Avaya™ Interaction Center

Release 6.0

Telephony Services for Aspect CallCenter

DXX-1035-01
Issue 2.0
June 2002

© **2002, Avaya Inc.**
All Rights Reserved

Notice
Every effort was made to ensure that the information in this book was complete and accurate at the time of printing. However, information is subject to change.

Preventing Toll Fraud
"Toll fraud" is the unauthorized use of your telecommunications system by an unauthorized party (for example, a person who is not a corporate employee, agent, subcontractor, or working on your company's behalf). Be aware that there may be a risk of toll fraud associated with your system and that, if toll fraud occurs, it can result in substantial additional charges for your telecommunications services.

Avaya Fraud Intervention
If you *suspect that you are being victimized* by toll fraud and you need technical support or assistance, call Technical Service Center Toll Fraud Intervention Hotline at +1 800 643 2353.

Providing Telecommunications Security
Telecommunications security (of voice, data, and/or video communications) is the prevention of any type of intrusion to (that is, either unauthorized or malicious access to or use of your company's telecommunications equipment) by some party.

Your company's "telecommunications equipment" includes both this Avaya product and any other voice/data/video equipment that could be accessed via this Avaya product (that is, "networked equipment").

An "outside party" is anyone who is not a corporate employee, agent, subcontractor, or working on your company's behalf. Whereas, a "malicious party" is anyone (including someone who may be otherwise authorized) who accesses your telecommunications equipment with either malicious or mischievous intent.

Such intrusions may be either to/through synchronous (time-multiplexed and/or circuit-based) or asynchronous (character-, message-, or packet-based) equipment or interfaces for reasons of:

- Utilization (of capabilities special to the accessed equipment)
- Theft (such as, of intellectual property, financial assets, or toll-facility access)
- Eavesdropping (privacy invasions to humans)
- Mischief (troubling, but apparently innocuous, tampering)
- Harm (such as harmful tampering, data loss or alteration, regardless of motive or intent)

Be aware that there may be a risk of unauthorized intrusions associated with your system and/or its networked equipment. Also realize that, if such an intrusion should occur, it could result in a variety of losses to your company (including but not limited to, human/data privacy, intellectual property, material assets, financial resources, labor costs, and/or legal costs).

Your Responsibility for Your Company's Telecommunications Security
The final responsibility for securing both this system and its networked equipment rests with you - an Avaya customer's system administrator, your telecommunications peers, and your managers. Base the fulfillment of your responsibility on acquired knowledge and resources from a variety of sources including but not limited to:

- Installation documents
- System administration documents
- Security documents
- Hardware-/software-based security tools
- Shared information between you and your peers
- Telecommunications security experts

To prevent intrusions to your telecommunications equipment, you and your peers should carefully program and configure your:

- Avaya-provided telecommunications systems and their interfaces
- Avaya-provided software applications, as well as their underlying hardware/software platforms and interfaces
- Any other equipment networked to your Avaya products.

Avaya National Customer Care Center
Avaya provides a telephone number for you to use to report problems or to ask questions about your contact center. The support telephone number is 1-800-242-2121.

Ordering Information
Avaya Publications Center
Voice: +1 800 457 1235
International Voice: 410 568 3680
Fax: +1 800 457 1764
International Fax: 410 891 0207
Email: totalware@gwsmail.com

Write: GlobalWare Solutions
Attention: Avaya Account Manager
200 Ward Hill Avenue
Haverhill, MA 01835 USA

Order: Document No. DXX-1035-01, Issue 2.0, June 2002

To order product documentation online, go to http://www.avayadocs.com, click on Online Services, and select the appropriate product group.

Warranty
Avaya Inc. provides a limited warranty on this product. Refer to the "Limited Use Software License Agreement" or other applicable documentation provided with your package to establish the terms of the limited warranty.

Avaya Web Page
http://www.avaya.com

Trademarks
Avaya, Conversant, CustomerQ, Definity, DefinityOne, Nabnasset, Quintus, and WebQ are registered trademarks or trademarks of Avaya Inc. in the United States or other countries or both.

Portions of Avaya Interaction Center include technology used under license as listed below, and are copyright of the respective companies and/or their licensors:

ActivePerl is a trademark of ActiveState Tool Corp. This product includes software developed by the Apache Software Foundation (http://www.apache.org/). Cognos, Impromptu and Powerplay are registered trademarks of Cognos Incorporated. YACC++ is a registered trademark of Compiler Resources, Inc. APEX, ComponentOne, VideoSoft, True DBGrid, VSVIEW, SizerOne, VS-OCX, VSFlexGrid, VSFORUM, VSREPORTS, VSDOCX, VSSPELL, and TrueDBList are either registered trademarks or trademarks of ComponentOne LLC. CT Connect, Dialogic, Intel, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries. Hummingbird is a registered trademark of Hummingbird, Ltd. SearchServer is a trademark of Hummingbird, Ltd. RISC System/6000 and DirectTalk/2 are trademarks of International Business Machines Corporation in the United States or other countries or both. IBM, OS/2, AS/400, CICS, WebSphere, CT, VisualAge, and DirectTalk are registered trademarks of International Business Machines Corporation in the United States or other countries or both. Lotus and Lotus Sametime are trademarks or registered trademarks of Lotus Development Corporation and/or IBM Corporation in the United States, other countries, or both. VisualX is a registered trademark of Intergroup Technologies, Inc. ActiveX, Visio, Internet Explorer, Windows, Windows NT, Windows 2000, Win32s, SQL Server, Visual Basic, Visual C++, Outlook, and FrontPage are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. TimesTen is a registered trademark of TimesTen Performance Software. Oracle is a registered trademark, and Oracle8i and Oracle® SQL/Services are trademarks or registered trademarks of Oracle Corporation. Rogue Wave and .h++ are registered trademarks of Rogue Wave Software Inc. SourcePro is a trademark of Rogue Wave Software, Inc. Siebel is a trademark of Siebel Systems, Inc. BasicScript is a registered trademark of Summit Software Company. Sun, iPlanet, Java, Solaris JRE, J2EE, JavaServer Pages, and all Java-based trademarks are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both. SPARC is a registered trademark of SPARC International, Inc. Products bearing SPARC trademarks are based on an architecture developed by Sun Microsystems, Inc. In3D is a trademark of Visual Insights, Inc. InstallShield® is a registered trademark and service mark of InstallShield Software Corporation in the United States and/or other countries. ORBacus is a trademark of IONA Technologies PLC. Formula One is a licensed trademark and Tidestone Technologies, Inc. Visual Components, First Impression, and VisualSpeller are registered trademarks of Tidestone Technologies, Inc. JRun is a trademark of Macromedia, Inc. in the United States and/or other countries. Intervoice is a registered trademark of Intervoice-Brite, Inc. UNIX is a registered trademark of The Open Group in the United States and other countries. Acrobat is a registered trademark of Adobe Systems.

Other product and brand names are trademarks of their respective owners.

Acknowledgment
This document was written by the CRM Information Development group of Avaya

CONTENTS

BEFORE YOU BEGIN 7

1 OVERVIEW 11

What is the Legacy Aspect TS? 11
Example Call Flow 12
Aspect CallCenter Requirements 15
TS Behavior During Hardphone/Softphone Interaction 15

2 TRACKING CALL AND AGENT DATA 17

How Call Information is Tracked 17
Call Container Contents 18
Reporting from Call Containers 19
Agent Containers 20

3 SETTING UP THE ASPECT CALLCENTER 23

Data InterLink Records 23
Overview of System Interaction 25
Trunk Records and Trunk Group Records 26
Call Control Tables 27
User Records 28
Class of Service Records 29
The Send Data Command 30
Log File Entries 31

4 CONFIGURING THE TELEPHONY SERVER 33

Configuration Overview 33
Agent Monitoring 35
Agent Configuration 35

TCP/IP Configuration 36
Using the TS in a WAN Environment 36
How the TS Interacts with the Workflow Server 36

5 IDL AND METHOD DESCRIPTIONS 39
Background Information 39
Interface Definition Language (IDL) 40
List of Generic TS Methods 41
Descriptions of Generic TS Methods 42

6 EVENTS 61
List of Events 61
Event Descriptions 62

7 ALARMS 67
List of Alarms 67

8 CALL FLOWS 69
Route Call 69
Inbound Call 70
Outbound Call 71
Busy Destination 72
Blind Transfer 73
Consultative Transfer 74
Internal Call 75

A SAMPLE CALL CONTROL TABLES 77
Default MakeCall CCT 77
Default Transfer CCT 78

B SEND DATA COMMAND SUBTYPES 81

INDEX 89

BEFORE YOU BEGIN

Typographical Conventions

This guide uses the following font conventions:

Font Type	Meaning
`code`	This font signifies commands, information that you enter into the computer, or information contained in a file on your computer.
italics	This font is used to add emphasis to important words and for references to other chapter names and manual titles. It also indicates variables in a command string.
jump	Blue text in online documents indicates a hypertext jump to related information. To view the related material, click on the blue text.

Notes, Tips, and Cautions

Note: A note calls attention to important information.

Tip: A tip offers additional how-to advice.

Caution: A caution points out actions that may lead to data loss or other serious problems.

Contacting Technical Support

If you are having trouble using Avaya software, you should:

1. Retry the action. Carefully follow the instructions in written or online documentation.
2. Check the documentation that came with your hardware for maintenance or hardware-related issues.

3 Note the sequence of events that led to the problem and the exact messages displayed. Have the Avaya documentation available.

4 If you continue to have a problem, contact Avaya Technical Support by:

- Logging in to the Avaya Technical Support Web site (http://www.avaya.com/support/qq).
- Calling or faxing one of the following numbers from 8:30 a.m. to 8:30 p.m. (Eastern Standard Time), Monday through Friday (excluding holidays):
  - Toll free in the U.S. only: 1-888-TECH-SPT (1-888-832-4778)
  - Direct line for international and domestic calls: 512-425-2201
  - Direct line for faxes: 512-719-8225
- Sending email with your question or problem to `crmsupport@avaya.com`. You may be asked to email one or more files to Technical Support for analysis of your application and its environment.

Note: If you have difficulty reaching Avaya Technical Support through the above URL or email address, please go to www.avaya.com for further information.

Product Documentation

Most Avaya product documentation is available in both printed and online form. However, some reference material is available only online, and certain information is available only in printed form. A PDF document with detailed information about all of the documentation for the Avaya Interaction Center is included in the `Doc` directory on the product CD-ROM. This PDF document is also included on the separate documentation CD-ROM.

Readme File

The Readme file is an HTML file included on the Avaya Interaction Center software CD-ROM. This file contains important information that was collected too late for inclusion in the printed documentation. The Readme file can include installation instructions, system requirements, information on new product features and enhancements, suggested work-arounds to known problems, and other information critical to successfully installing and using your Avaya software. You may also receive a printed Addendum to the Readme, containing similar information uncovered after the manufacture of the product CD-ROM. You should review the Readme file and the Readme Addendum before you install your new Avaya software.

Electronic Documentation

The electronic documentation (in PDF or HTML format) for each Avaya Interaction Center product is installed automatically with the program. Electronic documentation for the entire Avaya product suite is included on the product CD-ROM and the documentation CD-ROM.

You can also view the documentation set online at http://www.avayadocs.com.

Printed Documentation

You can purchase printed copies of these manuals separately. For details, see "Ordering Information," on the back of this manual's title page.

License to Print the Electronic Documentation

Online copies of documentation are included on the CD-ROM that accompanies every software release. An Avaya customer who has licensed software (a "Licensee") is entitled to make this online documentation available on an internal network or "intranet" solely for the Licensee's use for internal business purposes. Licensees are granted the right to print the documentation corresponding to the software they have purchased solely for such purposes.

Right-To-Print License Terms

Documents must be printed "as-is" from the provided online versions. Making changes to documents is not permitted. Documents may be printed only by any employee or contractor of Licensee that has been given access to the online documentation versions solely for Licensee's internal business purposes and subject to all applicable license agreements with Avaya. Both online and printed versions of the documents may not be distributed outside of Licensee enterprise or used as part of commercial time-sharing, rental, outsourcing, or service bureau use, or to train persons other than Licensee's employees and contractors for Licensee's internal business purposes, unless previously agreed to in writing by Avaya. If Licensee reproduces copies of printed documents for Licensee's internal business purposes, then these copies should be marked "For internal use only within <Licensee> only." on the first page or cover (where <Licensee> is the name of Licensee). Licensee must fully and faithfully reproduce any proprietary notices contained in the documentation. The copyrights to all documentation provided by Avaya are owned by Avaya and its licensors. By printing any copy of online documentation Licensee indicates its acceptance of these terms and conditions. This license only governs terms and conditions of printing online documentation. Please reference the appropriate license agreement for terms and conditions applicable to any other use, reproduction, modification, distribution or display of Avaya software and documentation.

Educational Services

Avaya University provides excellent training courses on a variety of topics. For the latest course descriptions, schedules, and online registration, you can get in touch with us:

- Through the web at http://www.avaya-learning.com
- Over the telephone at 800-288-5327 (within the U.S.) +001 303-406-6089 (outside of the U.S.)
- Through email at `Avaya.U.Helpdesk@accenture.com`

CHAPTER 1
OVERVIEW

Computer Telephony Integration (CTI) involves integrating telephone hardware (such as IVRs and ACDs) with computers, and software. The Avaya Computer Telephony Server (TS) is at the core of the Avaya Computer Telephony solution for Avaya™ Interaction Center 6.0.

There are two Avaya Computer Telephony Servers that support the Aspect CallCenter ACD.

- The Legacy Aspect TS, documented in this manual, is the original server which was written specifically for the ACD and is supported on Aspect CallCenter version 7.2 for Avaya Interaction Center (Avaya IC) 6.0.
- The Aspect TS, which can be configured for other ACDs, supports Aspect CallCenter 8.0 in this release. Refer to the the *Telephony Connectors Programmer's Guide* for more information about this TS.

What is the Legacy Aspect TS?

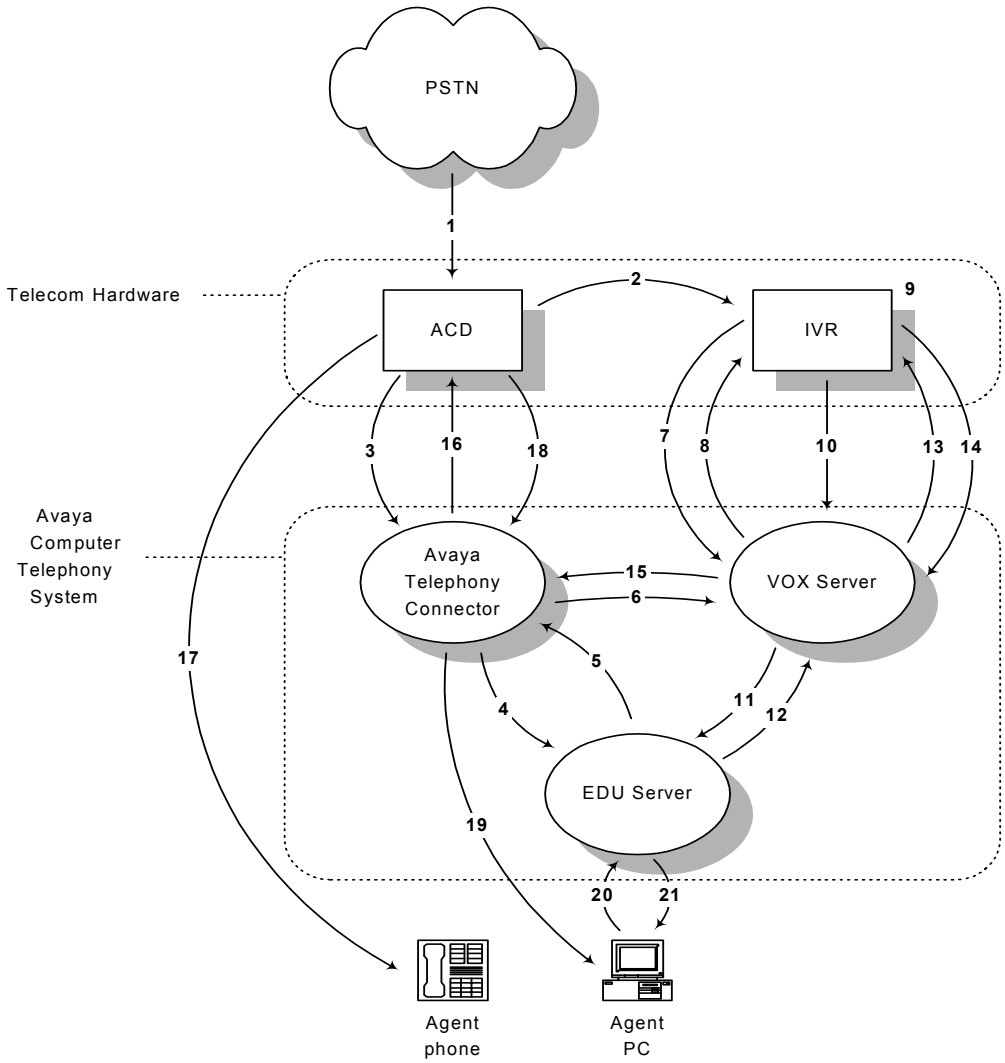
The Legacy Aspect TS is an Avaya software component responsible for linking Avaya Interaction Center (Avaya IC) to CTI products such as private branch exchange (PBX) systems, automatic call distribution (ACD) systems, and interactive voice response units (IVRs).

Using the Legacy Aspect TS, the other servers and clients in the Avaya Computer Telephony environment request telephony services (for example, transfer a call, route a call, hang up a call) by invoking TS methods. Clients and servers also "assign" to the TS to receive telephone-related events such as "there is a call from customer 999-1111".

The Legacy Aspect TS insulates client applications from the switch and its interface by encapsulating switch-specific features. Client applications are thus portable across multiple types of switches, CTI links, and other types of telephony interfaces, such as ISDN. Multiple Aspect Telephony Servers can be implemented for redundancy to enable transparent failover in case the ACD link, server network, or server software fails.

Example Call Flow

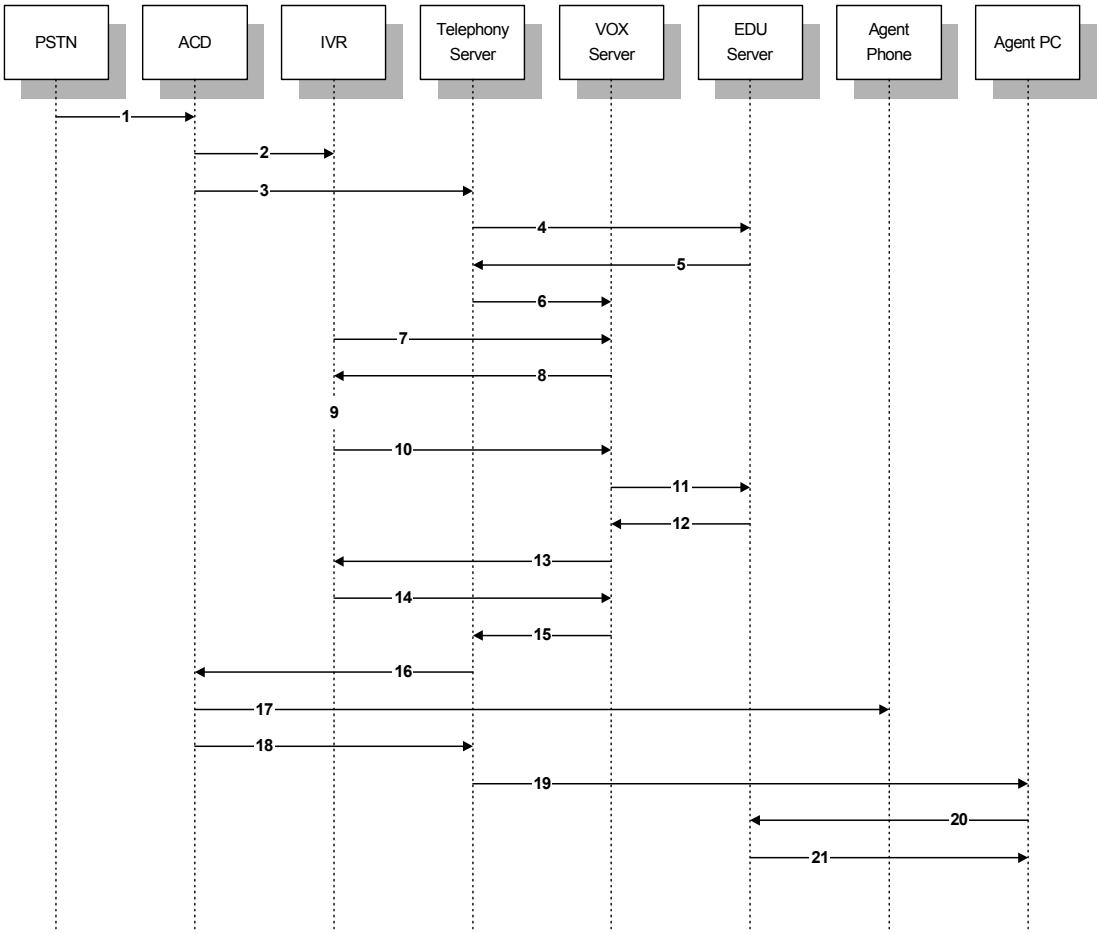
The function of the Legacy Aspect TS can best be understood by examining a typical call flow. Note that call flows vary for each installation. Additional call flow diagrams are presented in “Call Flows,” on page 69.



Step	Description
1	A call arrives at the switch (PBX) from the public network (“cloud”).
2	The switch, in this example, is set up to route calls directly to an Interactive Voice Response Unit (IVR). The call moves from the PBX to the IVR.
3	The PBX notifies the Legacy Aspect TS that it has received a call on a line in which the server had previously indicated an interest.

Step	Description
4	The Legacy Aspect TS invokes a method (VDU.Create) on the EDU Server, which instructs the EDU Server to create a new Electronic Data Unit (EDU) for the call. The method specifies a list of elements to place into the EDU (such as ANI and DNIS).
5	The EDU Server creates the EDU with the specified elements and sends it to the Legacy Aspect TS.
6	The Legacy Aspect TS checks its tables to see which clients or servers previously indicated an interest in monitoring events on the phone line to which the call was routed. The Legacy Aspect TS discovers that the VOX Server is an interested party, so the Legacy Aspect TS sends an incoming call event to the VOX Server. The event includes the extension to which the call was routed and the call's EDU identifier (EDUID).
7	The IVR notifies the VOX Server of the new call. The notification includes the channel number on which the call arrived.
8	The VOX Server associates the IVR channel number with a telephone extension. The VOX Server then matches that extension with the incoming call event received in step 6. The EDUID from the incoming call event is then returned to the IVR.
9	The IVR greets and prompts the caller for an account number and collects the digits entered by the caller via a touch-tone key pad.
10	The IVR instructs the VOX Server to place the account number in the EDU as a name/value pair (e.g., "account", "123456789").
11	The VOX Server translates the IVR request into a CORBA method call to the EDU Server (VDU.SetOneValue).
12	The EDU Server stores the value and returns a response to the VOX Server (VDU.SetOneValue.response).
13	The VOX Server informs the IVR that its request (see step 10) has been granted.
14	The IVR invokes a method on the VOX Server (VOX.Transfer) to route the call to an agent queue.
15	The VOX Server invokes the a transfer method (TS.Transfer) on behalf of the IVR.
16	The Legacy Aspect TS instructs the PBX to transfer the call.
17	The PBX places the call in a queue and transfers it to the next available agent, whose telephone rings.
18	The PBX notifies the Legacy Aspect TS of the transferred call. The Legacy Aspect TS keeps track of which EDUID is associated with the call.
19	The Legacy Aspect TS sends an incoming call event (TS.IncomingCall.event) to the client application assigned to the extension to which the call was transferred. The event includes the EDUID associated with the call.
20	The client application uses the EDUID to request an account ID from the EDU Server (VDU.GetOneValue).
21	The EDU Server responds to the client application (VDU.GetOneValue.response) with the requested account ID.

The following diagram uses a different method to illustrate the same call flow.



Aspect CallCenter Requirements

In order for the Legacy Aspect TS to work with Aspect CallCenter, the following Aspect CallCenter releases, versions, and options are required:

- Aspect CallCenter, release 6.*x* or 7.2 (releases 7.0 and 7.1 are excluded)
- Aspect CallCenter Application Bridge, version 4
- Aspect CallCenter Event Bridge option

TS Behavior During Hardphone/Softphone Interaction

The following table depicts those interactions between hard and soft phone sets. The hardphone is the physical phoneset sitting on an agent's desk, while the softphone is the software-implemented phone depicted by a GUI on an agent's computer screen.

This interaction in a CTI environment is strongly dependent on events being passed to the telephony server from the PBX, and the telephony server in turn passing those events through to the client, to keep the hardphone and the softphone in synch. As such, certain limitations are imposed by the hardware involved, and not on the software implementation. Additionally, certain keys available on the hardphone set are there as controls specific to that particular phone set. Some of these keys may be function programming keys, dial last number keys, etc. These keys do not require PBX interaction, and as such do not generate events, and are completely transparent to the CTI software.

The following interaction table attempts to identify those telephony features that are supported or not supported by the Avaya Computer Telephony at this time for the Aspect CallCenter switch. This table compares event results between the physical phoneset and the softphone.

Action	Aspect CallCenter	
	Hardphone	Softphone
alternate call	yes*	yes
answer call, connect	yes	yes
auto ready (auto-in)	no	no
blind transfer	yes*	yes
bridge two calls	yes	no
busy (aux-work)	yes*	yes*
conference call	no	yes
consultative transfer	yes*	yes
consultative transfer one-step	no	no
forward calls, cancel forward calls	no	no
hang up call	yes	yes

Action	Aspect CallCenter Hardphone	Softphone
hold call	yes*	yes*
listen disconnect/reconnect (mute)	no	no
login	yes*	yes*
logout	yes*	yes*
make call	yes	yes
make call predictive	no	future
message waiting, cancel msg wait	no	no
monitor and control DN	no	yes
monitor DN	no	no
monitor queue	no	no
network take back and transfer	no	no
off hook	no	no
on hook	no	no
ready (in service/manual-in)	yes*	yes*
reconnect held call	yes*	yes
redirect alerting call	no	no
route call	no	yes
send all calls, cancel send all calls	no	no
send DTMF tones	no	no
single step conference	no	no
third party drop	yes	yes
walk away, return fr. walk away	no	no
wrap up (after-call-wk)	yes*	yes*

* == Requires Event Bridge Option and Application Bridge 4 release 6

CHAPTER 2
TRACKING CALL AND AGENT DATA

The Legacy Aspect TS automatically tracks two types of call data: the names of interested clients and call end point information. In addition, the Legacy Aspect TS tracks various types of agent activity for reporting purposes.

How Call Information is Tracked

Avaya Computer Telephony stores all information related to a telephone call in a call detail record known as an Electronic Data Unit (EDU). EDUs are maintained by the EDU Server. The Legacy Aspect TS ensures that the EDU follows its call as the call moves through the system. This prevents customers from having to repeat their name and other information, and eliminates redundant database queries.

EDU information can be organized into containers. A call container is a grouping of values under a common name. Call containers are hierarchical structures of data within an EDU. For more information about EDUs and call containers, see the *Electronic Data Unit Server Programmer's Guide*.

The various parties involved in a call, such as agents, callers, and IVR ports, are referred to as end points. Each time a call is connected to a new party, information is stored about the new end point. To manage this, the Legacy Aspect TS creates a call container for each end point. All EDU data relating to call end points (events and name/value pairs) are stored in call containers.

The first end point of a call is named voice.1, the second is voice.2, the third is voice.3 and so on. Information about each end point is stored in a subordinate structure (voice.2.phone, voice.2.ptype, voice.2.tran, etc.). A new end point is created each time a new client becomes associated with the EDU.

For performance reasons, the Legacy Aspect TS accumulates information about each end point until the connection is terminated. When the connection is terminated, the information is sent to the EDU Server.

Creation of call containers is controlled by the Create Containers (tscon) configuration parameter.

Note: Additional information about telephone call connections and end points should be stored in separate data structures, not in call containers.

If the Legacy Aspect TS and EDU Server are running on different machines, you must keep the clocks on the two machines synchronized. Several tools are available for synchronizing clocks, including the UNIX timed (time daemon) process, which synchronizes a group of UNIX servers to a master clock. See your operating system documentation for more information about clock synchronization tools.

Call Container Contents

The following table describes a call container in which end point events and attributes are stored. X represents the unique identification number for each end point. Y and Z represent sequence numbers within each end point's activities.

Name	Value	Explanation
voice.X	delta time	For voice.1, this is base time (should be zero). For other cases, it is elapsed time in seconds since creation of the EDU.
voice.X.abandon	reason for abandoning the call: "in queue," "while ringing," "while on hold."	If the exit reason of a call end point is "abandon," this item provides additional details.
voice.X.agent_key	unique database record key	The key that retrieves the agent record from the database.
voice.X.conferencedest.Z	phone number	The destination phone number (Z) of a conference call, where Z = 1, 2, 3, ...
voice.X.connect	delta time	The elapsed time in seconds between the creation of the EDU and the time the call was connected.
voice.X.destination	phone number	Phone number of client at end point X.
voice.X.direction	inbound/outbound	The direction of the call to or from the agent.
voice.X.exit_reason	reason for exit: "normal," "transfer," "abandon," "other."	The switch supplies a reason for the termination of a call end point.
voice.X.holdtime.Y	time in seconds	Time spent on hold during hold instance Y.
voice.X.leg_id	unique id (UUID)	Unique leg_id of the current leg of the call.
voice.X.loginid	loginid	Login id of client at end point X.
voice.X.origin	phone number	Phone number (ANI) at other end of end point X.

Name	Value	Explanation
voice.X.queue	delta time	The elapsed time in seconds between the creation of an EDU and the time at which the switch reports the call as queued.
voice.X.queue_number	queue number	Queue number configured within the CallCenter switch as an Application Group.
voice.X.queuetime	time in seconds	Time reported by the switch between a call arriving on an inbound trunk of the switch and the call being sent to an agent queue. **Note:** Currently, this value is always zero (0). Subsequent releases of the telephony server will supply more meaningful values.
voice.X.ringtime	time in seconds	Time reported by the switch between agent phone starting to ring and agent answering the phone.
voice.X.talktime	time in seconds	Time reported by the switch between agent answering the phone and agent hanging up the phone.
voice.X.transfer	phone number	The phone number to which the call has been transferred.

Reporting from Call Containers

State durations (connect duration, talk time, hold time, etc.) and event counts (number of times a call was placed on hold, etc.) are not calculated by the Legacy Aspect TS, but can be derived from the time stamps and other information in a TS Container.

The following table describes use of the process measurement data in a TS Container.

Measurement	Definition	Derivation
Hold Time	Time spent on hold during hold instance Y.	Derived from voice.X.holdtime.Y.
Queue Time	Time reported by the switch between a call arriving on an inbound trunk of the switch and the call being sent to an agent queue. TS Aspect-specific.	Value stored in voice.X.queuetime.

Measurement	Definition	Derivation
Ring Time	Time reported by the switch between agent phone starting to ring and agent answering the phone. TS Aspect-specific	Value stored in voice.X.ringtime.
Talk Time	The length of time that each agent was connected to the telephone call (including time on hold).	Value stored in voice.X.talktime.
Hold Count	The total number of times a call was placed on hold.	Total number of voice.X.holdtime.Y events for a given instance of X.

Agent Containers

As a call moves between agents in a contact center, each agent (person or an IVR) may wish to store information about the call in a way that will not interfere with the next or last agent who handled the call. To manage this, the Legacy Aspect TS can create an agent container.

Each agent involved in the call takes a private subtree of the agent container for his or her own use. The first agent uses the name agent.1, and creates names below it — agent.1.callduration, agent.1.reasoncode, and so on. The next agent uses the name agent.2 and sets his own values in that subtree — agent.2.callduration, and so on.

Reporting From Agent Containers

Whenever a contact is terminated, the Legacy Aspect TS updates information in agent container queues. This information is then used to generate reports about agent activity.

Data Recorded	Calculation
Number of contacts offered	A tally of items posted to a specific ADU.
Number of contacts handled	Derived from TS container end point 1. Number of contacts handled is a combination of voice.1 and voice.1.talktime where talk time is greater than a specific period of time (e.g., two seconds.)
Number of contacts abandoned while in queue, ringing, or on hold	Derived from TS container end points that include voice.abandon items. Data is recorded for each agent.
Average call handling time	The sum of all talk time, ring time, and queue time divided by the number of contacts handled.
Average talk time per contact	Total talk time divided by the number of contacts handled. Since the switch reports talk time from its own perspective, talk time includes the period in which a contact is on hold; to exclude time on hold, talk time should first be reduced by the following: voice.X.holdtime.Y (for all Ys in the container) divided by the number of contacts handled.

Data Recorded	Calculation
Average talk time per agent	Similar to calculation for average talk time per contact, except this one is specific to each TS container end point item voice.X.loginid, grouped by login ID. Data can be derived from the ADU by dividing total agent talk time by the number of agents.
Average wrap up time	Raw data is in the ADU itself. Every agent has an ADU after a successful voice.assign. After hanging up a call, the agent transitions to a wrap up state, and the time of this transition is recorded in the agent's ADU. The time of the agent's next state transition is also recorded in the ADU. So, wrap up time is the period between these two transition times. The average, therefore, is derived by dividing the total wrap up time for all calls by the total number of calls.
Average answer time	Total ring time for the number of contacts offered, divided by the number of contacts.
Average time before abandoning a call	Raw data is derived from TS container end points that include voice.abandon items. Data is recorded for each agent. Average time before abandoning a call is the sum of ring time, queue time, and talk time for all abandoned calls, divided by the number of abandoned calls.
Average number of transfers	Derived by dividing the number of contacts handled by the number of voice.X.transfer items in the container.
Number of times a contact is placed on hold	Raw data comes from the number of contacts handled. Calculation is as follows: number of voice.X.holdtime.Y items for all Ys within each X.
Duration of the current agent state	Raw data is in the ADU. For a given ADU, subtract the current time from the time the current agent state was recorded in the ADU.

CHAPTER 3
SETTING UP THE ASPECT CALLCENTER

This chapter describes how to set up and configure the Aspect CallCenter to integrate with Avaya Computer Telephony.

Data InterLink Records

When adding Avaya Computer Telephony to Aspect CallCenter, create a Data InterLink Record to define the attributes of the physical link between Aspect and Avaya Computer Telephony. If a Data InterLink Record does not exist or is not defined correctly, Aspect cannot communicate with Avaya Computer Telephony.

The Data InterLink Record is accessed through the Aspect menu option: Database > Resources > Data Interlink > Update Records.

Example Data Interlink Record

The following illustrates an example Data InterLink Record.

Update Data System InterLink Record

Data InterLink Number>	14		
Description:	Test		
Version Number>	4		
Backup Link>	0		
Physical Protocol>	2	Port	7002
Link Protocol>	3		
CallCenter Address:	AspTCP	Data System Address>	AspTCP
Message Format>	F		
Field Separator>	5		
Send Type>	Y		
Disconnect Notices>	Y	Subtype:	
Transfer Notices>	Y	Subtype:	
Character Set>	A		

Update Data System InterLink Record			
Receive Data Timeout:	10	Monitor Host>	Y Interval: 10
Modify [M] or Quit [Q]?			

This record contains several fields of particular significance to Avaya Computer Telephony.

Field	Required Value
Data InterLink Number	Link numbers 11 through 30 indicate an Ethernet connection.
Version Number	Application Bridge version used by the primary link: 1 = Aspect CallCenter release 4.2 2 = Aspect CallCenter releases 4.3 and 4.4 3 = Aspect CallCenter release 5.x 4 = Aspect CallCenter releases 6.x and 7.2
Backup Link	Link number used as a backup if the primary link fails. Avaya Computer Telephony does not use a backup link; enter 0.
Physical Protocol	Enter 2 for Ethernet.
Port	Port number defined by your Aspect service representative. The port number and data system address determine which TS will process requests sent over this link. This field maps to the "Device (link1)" TS configuration parameter (see page 34).
Link Protocol	Enter 3 for TCP/IP.
CallCenter Address Data System Address	Currently, these two fields must contain identical values and must not be empty (see Note below). For each Ethernet connection, your Aspect service personnel will define an Application Bridge node number, host name, and data system address. The choices available for the data system address in the interlink record are determined by the data system addresses entered by your Aspect service personnel. **Note:** The Aspect documentation states that the CallCenter Address and Data System Address fields must contain different values. A future release of the TS will comply with this specification.
Message Format	Variable (V) is recommended. If using Fixed format, ensure that the line number configuration in your IVR includes leading zeroes to accommodate the number of digits in your dial plan. If changing from Fixed to Variable, you must stop and restart the Legacy Aspect TS after making the change.
Field Separator	Must be 5 (comma).
Send Type	Must be Y.
Disconnect Notices	Must be Y.

Field	Required Value
Transfer Notices	Must be Y.
Character Set	Must be A (ASCII).
Receive Data Timeout	10 or 20 is standard.
Monitor Host	Set to Y.

Note: Do not modify the Data InterLink Record while agents are logged in.

Overview of System Interaction

The Aspect CallCenter system interacts with Avaya Computer Telephony through a series of commands. The commands are sent to Avaya Computer Telephony via the Application Bridge to the Legacy Avaya TS, which in turn forwards the requests to the Avaya Workflow Server.

The following is a brief description of the flow of information that occurs when a call is handled by the Aspect CallCenter and Avaya Computer Telephony. Additional information is provided in subsequent sections.

1. Call arrives on a trunk.
2. Trunk Table and Trunk Group Records are read.
3. CCT is executed.
4. Send Data command initiates communication with the Legacy Aspect TS.
5. Receive Data command gathers information from the Legacy Aspect TS.
6. Call is routed to an agent group and subsequently to an agent.
7. Agent's User Record is read.
8. Class of Service Record is read to determine agent's auto answer and wrapup settings.

Note: This chapter describes only those items of particular importance to the Legacy Aspect TS. For a complete description of the records and tables used with the Aspect CallCenter, refer to the *Aspect CallCenter Workstation User's Guide* and the *Application Bridge Implementation Guide*. Items such as link numbers, table numbers, and announcement numbers used in the sample records are for example only. You must substitute settings appropriate for your environment.

Trunk Records and Trunk Group Records

When a call arrives at the Aspect CallCenter, the Trunk Record for the call's trunk is read.

An example Trunk Record is shown below.

Update Trunk Record

Trunk Number>	4		
Circuit ID:	11AX57290		
Circuit Provider:	PACBELL		
Directory Number:	7872800		
Test Number:	7872800		
Card Number>	19 Loop/Ground Start Trunk		
Port Number:	4		
Offline Instrument Connection>	0		
Trunk Type>	L		
Direction>	B		
Incoming Group Number>	23 CCT 69 ->LNK 14		Errors
Outgoing Group Number>	5 Outbound	Noise:	0
Supervision>	L	Echo:	0
Digit Characteristics>	DTMF	Volume:	0

Modify [M] or Quit [Q]?

One of the items defined in this record is the Incoming Group Number, which identifies the trunk group associated with this trunk. In the example above, the record for trunk group 23 would be read.

The Trunk Group Record identifies the Call Control Table (CCT) to be used to control calls arriving on this trunk group. In the following example, CCT 69 is identified.

Update Trunk Group Record

Trunk Group Number>	23
Trunk Group Name:	CCT 69 -> LNK 14
Priority:	9
Call Type Announcement Number>	0
Call Control Table Number>	69 DNIS->LINK 14
Number of DNIS Digits:	0
Capable of Outgoing Flash>	Y
Capable of Incoming Flash>	Y
ISDN Service Number>	1 MEGACOM

WARNING! To initiate a change in the DNIS field,
associated trunks must be deactivated and then activated.

Modify [M] or Quit [Q]?

Call Control Tables

A CCT contains a series of steps to be executed in the control of a call. Following the example, the steps in CCT 69 would be executed. (Additional sample CCTs are presented in Appendix A.)

Update Call Control Table

	Table Number>	69	Description:	DNIS->LINK 14
	Application Number>	65	Description:	QA TESTING APP
Step #	Command	----------Attributes----------		54 % Full
1	MOVE	CONTENTS OF VARIABLE [#] TO VARIABLE [B]		
2	MOVE	CONTENTS OF VARIABLE [$] TO VARIABLE [A]		
3	SEND DATA	LINK# > 14 SUBTYPE startcall VAR LIST A-E ON ERROR, EXECUTE STEP # 13		
4	RECEIVE DATA	LINK# > 14 ON NACK, EXECUTE STEP # 5 ON ERROR, EXECUTE STEP # 13		
5	SELECT	> AGENT BY> GROUP NUMBER> 7 ON SUCCESS, EXECUTE STEP # 8		
6	QUEUE	30 SECONDS		
7	GOTO	STEP NUMBER: 5		
8	SEND CONNECT	LINK #14 SUBTYPE LINK14 VAR LIST A-E		
9	ANNOUNCEMENT	NUMBER> 68		
10	CONNECT	ON NO ANSWER, EXECUTE STEP # 15		
11	ANNOUNCEMENT	NUMBER> 66		
12	GOTO	STEP NUMBER: 16		
13	ANNOUNCEMENT	NUMBER> 67		
14	GOTO	STEP NUMBER: 16		
15	ANNOUNCEMENT	NUMBER> 71		
16	DISCONNECT			

Several of the steps in the sample CCT illustrate items of particular significance to the Avaya Computer Telephony system.

Steps	Commands	Description
1, 2	Move	For an incoming call, the # buffer contains the DNIS (Dialed Number Identification Service). The TS expects the DNIS to be stored in variable B. For outbound calls, the # buffer is invalid. For an incoming call, the $ buffer contains the ANI (Announced Number Identifier). The TS expects the ANI to be stored in variable A. For an outgoing call, the $ buffer contains the dialed digits.
3	Send Data	This command sends a Call Information Message to the TS over the specified link. Refer to the "The Send Data Command," on page 30 for a description of the Send Data command, and refer to "Data InterLink Records," on page 23 for information on defining link numbers.

Steps	Commands	Description
4	Receive Data	The TS sends a Call Information Message Response to the Aspect, returning any data or routing instructions gathered.
5	Select	Selects an agent by group number. Refer to "User Records," on page 28 for information on defining agent (user) records.
8, 9, 10	Send Connect, Announcement, Connect	The Send Connect command sends a Call Connect Message to Avaya Computer Telephony over the specified link. The Call Connect message provides information to the TS prior to call connection, indicating that a call has been handled, to what type of destination the call will be connected, and passing any applicable information in variables A through E. The Send Connect command should always be followed by an uninterruptable announcement to allow adequate time for data to be gathered for screen population.

User Records

Step 5 in the example CCT (shown above) selects an agent in group 7. Each agent's configuration parameters are defined in a User Record.

Update User Record

```
          Extension Number> 210
            User ID Number: 123456789
  Full Name (Last, First): Hillman Jessica
          Preferred Name: Jessica
            Type of User> A
             User Status> A
             Status Date: 021502

      Default:                Current:
Supervisor Team>  1  CUST SVC        > 1  CUST SVC
   Agent Group>  3  CUST SVC        > 3  CUST SVC
Class of Service>  65 CTI TELESET     > 1  CTI TELESET
```

One of the items defined in an agent's User Record is the agent's Class of Service. In this example, the agent's Class of Service record is identified as 65.

Class of Service Records

A Class of Service Record defines telephone capabilities for a group of users.

Update Class of Service Record

Class of Service Number> 65
Class of Service Name: Standard Teleset
Instrument Type> T

User Attributes:

Auto Answer> Y	Wrap-Up After Incoming> Y
Automatic Available> N	Automatic Callback> Y
Direct Trunk Selection> N	Speaker> Y
User-to-User Calls> Y	Message Prompts> A
Righ-Through Calls> 4	Wrap-Up After Outgoing> N
Incoming Calls> 4	Help and Messaging> N
Reason Codes/Idle> Y	Supervisor Calls> Y
Reason Codes/Sign-Off> Y	

Outbound Access:

International Calls> Y	Public Network Calls> Y
Private Network Calls> Y	Operator-Assisted Calls> Y
PBX Calls> Y	AMISA Send> N

The fields most applicable to the Legacy Aspect TS are the Auto Answer and Wrap-up After Incoming settings.

If you are using Aspect CallCenter release 6.x or 7.2 with Application Bridge 4, setting AutoAnswer to Y is recommended (but not required).

If you are using VTel, set the Wrap-up After Incoming field to Y. If agents are set to enter wrap-up after incoming, there must be no wrap-up script defined in the application associated with the CCT controlling the call. That is, the Script Number field in the Application Record must contain a zero (0). This allows Avaya Computer Telephony to receive a disconnect message as soon as the agent enters wrap-up.

The following is a sample Application Record:

Update Application Record

Application Number>	65
Application Name:	Credit Processing
Music on Hold>	N
Script Number>	0
Primary Group Type>	A
Primary Group Number for Reporting>	65 Credits
Agent Group Number for CallBack Msg>	65
Temp Value To Display at Cut-through>	
Time of Longest Call Waiting (min:sec!)	S:S
Service Level Time (min:sec):	S:S
Service Level Percent:	S
Ani Request:	A

Modify [M] or Quit [Q]?

The Send Data Command

The Send Data command in a CCT sends a Call Information Message to the Legacy Aspect TS over the specified link. The following is a sample Send Data command.

```
SEND DATA LINK #> 1 SUBTYPE startcall VAR LIST A-E
ON ERROR, EXECUTE CCT # 999
```

The Send Data command contains the following fields:

Field	Description
LINK #>	Data InterLink number, which identifies the physical link between the Aspect CallCenter and the TS. Valid entries are defined in Data System InterLink Records. Refer to “Data InterLink Records,” on page 23 for additional information about Data System InterLink Records.
SUBTYPE	If used, contains a designator of up to 12 characters that provides additional information about the data being sent to the TS. Subtype of "startcall" initiates a new call if Variable E is blank. Refer to Appendix B for more information on the SUBTYPE field.
VAR LIST	If used, tells the switch to send the contents of one or more CCT Data Variables A - E in the Call Information Message's Data field. For an incoming call, the variables A and B are used to hold the ANI and DNIS. Variable E is reserved by Avaya Computer Telephony to hold the EDUID. The list can be a single variable, a list of variables, and/or a range of variables, such as: A A,C A-E A-C,E
ON ERROR, EXECUTE	If used, specifies the path to take if an error is encountered in processing the SEND DATA step. This path can be either another CCT or a step within the current CCT.

When the Legacy Avaya TS receives a Call Information Message (001, Send Data), it sends a TS.SendData.event to any client assigned with a *t criteria. The client would then send a TS.ReceiveData method request to the Legacy Avaya TS, and the Legacy Avaya TS would send an ACK back to the CallCenter, allowing the script to continue. Note that subtypes that specify "wait for data" (those with Return values) require a response. The TS.SendData.event is not sent in response to call information messages with a subtype of newtrans, startcall, gv*, sv* (with the exception of svlanguage), ev*, or ed*.

For detailed information on SendData command subtypes, see Appendix B.

Log File Entries

Log files can contain information on the messages passed between the Aspect CallCenter and the Legacy Aspect TS. Understanding log file entries can assist you in understanding the interaction between these systems. The content of log files is determined by the tracing level established in the Legacy Avaya TS's configuration file. Refer to *IC Administration Volume 1: Servers & Domains* for additional information on setting trace levels and on log file interpretation.

The following is a sample log file entry for a Call Information Message.

```
  1                    2    3        4    5  6 7 8   9 10  11

Info /From Aspect sun1, 001,startcall,293,T,23,,6699, , ,sun1:8006
[aspect_methods.c:711]
```

Callout	Value	Description
1	Header	Identifies the Aspect CallCenter address. It is mandatory for Ethernet connections.
2	Type	The three-digit identifier for this message type. (A Send Data command is a Call Information Message, which is type 001.)
3	Subtype	If specified in the CCT, the subtype provides additional information about this message. See Appendix B for more information.
4	Call ID	The identifier assigned to this call by Aspect CallCenter.
5	Orig Type	Identifies the type of originating equipment. Values are: I (Instrument Teleset), S (Station), T (Trunk), and O (incoming interqueue request).
6	Orig	Identifies the resource type index originating the call. Values range from 1 to 1200, depending on the originating equipment.
7	Data A	If ANI information has been moved from the $ buffer, it is stored in data variable A.

Callout	Value	Description
8	Data B	If DNIS information has been moved from the # buffer, it is stored in data variable B.
9, 10	Data C - E	If data is stored in these variables, it is displayed. Variable E is reserved for the EDUID.
11	UNIX or NT info	UNIX or NT machine name and port number of the Legacy Avaya TS being used.

The following is a sample log entry for a call information message response. Numbered items indicate differences from the call information message log entry.

```
     1                2     3    4     5      6    7                      8
Info/Aspect out sun1,  601,  293,  N,  000,  ,  n,  ,  sun1:8006 [aspect_methods.c:772]
```

Callout	Value	Description
1	Type	The three-digit identifier for this message type. A call information message response is type 601.
2	Call ID	The identifier assigned to this call by the CallCenter.
3	Resp	Contains a value of A (ACK) for a positive acknowledgment to the Call Information Message, or a value of N (NAK) for a negative acknowledgment. This allows the CCT to branch using the ON NAK statement after the RECEIVE DATA statement.
4	CCT	Specifies the CCT to which to transfer control when this message is received. Values are 000 through 500. This field should always be set to 000 to indicate that the current control table should be used. This field is ignored if Resp is N.
5 through 8	Data A - E	Contain data to be passed to the Aspect. Variable E is reserved for the EDUID.

CHAPTER 4
CONFIGURING THE TELEPHONY SERVER

This chapter contains configuration information for the Legacy Aspect TS as implemented for the Aspect CallCenter, version 7.2.

Configuration Overview

The Legacy Aspect TS is configured using IC Manager. The following table lists the label displayed on IC Manager, followed in parentheses by the parameter name as it is required internally by the Legacy Aspect TS.

For information on using IC Manager to set configuration parameters, refer to *IC Administration Volume 1: Servers & Domains*.

Label	Description
ACD Name (acdname)	Name of the ACD that this TS is serving. Provides a pick list of names assigned to the ACD during system configuration. Select the name assigned to the Aspect CallCenter 7 switch.
ACD Type (pbx_type)	Type of ACD with which the TS will communicate. Select Aspect.
ACD Model (pbx_model)	Model of Aspect ACD that corresponds to the selected ACD Type. Select Aspect.
ACD Protocol (pbx_protocol)	The protocol to be used between the TS and the ACD. Select Legacy.
Site (site)	The “Site” to which the TS is associated. The TS also uses this information to retrieve queues against which it performs internal monitoring.
Enable Call Containers (tscon)	Enables call containers for the TS if checked. Default is checked (TS call containers enabled).
Enable Agent Containers (aducon)	If checked, the ADU containers for the TS are turned on. Default is checked (agent containers enabled).
Use 5.6 State Fields (containers_56_style)	If checked, determines which containers should have entries in the 5.6 style. Default is unchecked.

Label	Description
Use 6.0 State Fields (containers_60_style)	If checked, determines which containers should have entries in the 6.0 style. Default is checked.
Wrap up by Client (wrapup_by_client)	If checked, the Telephony Server waits for the wrap up process to be completed before removing the call information from memory.
Wrap up Client Time to Live (min) (wrapup_client_ttl)	The period of time, in minutes, that the Telephony Server waits for the wrap up process to be completed by the client. If unchecked, the TS issues a request for wrap up on behalf of the client.
Wrap up Server Time to Live (min) (wrapup_server_ttl)	The period of time, in minutes, that the Telephony Server waits for the wrap up process to be completed by the server. If unchecked, the TS issues a request for wrap up on behalf of the server.
ACD Link (link1)	The device through which the TS communicates with the ACD. Enter the IP address of the MAPD card set. Maximum length is 32.
Call Plan (callplan)	Minimum number of digits to be entered for a valid extension number. Default is 1.
Device (link1)	Device through which the Legacy Aspect TS communicates with the switch, as defined by Aspect service personnel. Specifically, it is the TCP/IP "Port" number specified in the Data InterLink Record.
Connection Type (linktype)	Type of connection. Default is "tcp".
ACD Version (pbxver)	The version of the Aspect CallCenter ACD. Use the default 4.
Monitor Agent Group (AG)	The identifier of the agent group to be monitored. Default is 1.
Monitor All Agent Groups (AAG)	If checked, monitor all agent groups. Default is unchecked.
Monitor Super Agent Group (ASG)	The identifier of the super agent group to be monitored. Default is 1.
Monitor All Trunk Groups (ATG)	If checked, monitor all trunk groups. Default is unchecked.
Make Call CCT (makecallcct)	Default CCT to be used when an agent performs the MakeCall command. If not set, an alarm will be raised. The alarm can be ignored if calls will not be made or transferred by name.
Transfer CCT (txfercct)	Default CCT to be used when an agent performs a transfer command. If not set, an alarm will be raised.
Transfer Init CCT (txferinitcct)	Default CCT to be used when an agent performs a transfer init command. If not set, an alarm will be raised and this value will default to the Transfer CCT (txfercct) value. The alarm can be ignored if calls will not be made or transferred by name.

Agent Monitoring

Agent group configuration in the switch must be kept synchronized with IC Manager's agent definitions in the Directory Server for queue assignments. This must be done manually. The switch cannot automatically communicate its configuration information to Avaya IC, nor vice versa.

In the following table of monitoring parameters, which are also included in the "Configuration Overview," on page 33, note that ATG and AAG put maximum load on the link. If ATG or AAG are set to true, all of the trunks or agents in the system are monitored. Monitoring starts after the first heartbeat.

Label	Description
Monitor all agent groups (AAG)	If set to true, monitor all agent groups.
Monitor agent group (AG)	Monitor the specified agent group.
Monitor super agent group (ASG)	Monitor the specified super agent group.
Monitor all trunk groups (ATG)	If set to true, monitor all trunk groups.
Monitor trunk group (TG)	Monitor the specified trunk group.

Agent Configuration

The following agent configuration parameters are specified when administering an agent on Avaya IC using IC Manager. The table lists the label that appears in IC Manager, followed in parentheses by the parameter name as it is required internally by the Legacy Aspect TS.

For more information, refer to *IC Administration Volume 1: Servers & Domains*.

Label	Description
Phone ID (phone)	Agent number or a CCT number ("cctxxx"). Only ACD agents are supported on the Aspect.
Password (phone_passwd)	Password specified on Aspect CallCenter switch.
Phone Type (phone_type)	Type of phone. Must be set to ACD for Aspect users.
Equipment (equipment)	The number the switch uses to identify the equipment. Must be four digit, zero filled, beginning on the left (e.g., for 2 the field should contain 0002).

TCP/IP Configuration

Correctly defining the Data InterLink Record is critical to communication between Avaya Computer Telephony and the Aspect CallCenter. Refer to "Data InterLink Records," on page 23 for detailed information on the Avaya Computer Telephony requirements for the Data InterLink Record fields.

If problems arise with a TCP/IP connection, ensure that the port you have selected is not used by anyone else. To do this, use the command `netstat -a`.

Using the TS in a WAN Environment

In order to use the Legacy Aspect TS in a WAN environment, the Aspect CallCenter Network Inter-queue feature is required.

In a WAN environment, calls are passed between contact centers using the D channel of an ISDN line. The Call Control Table contains an Interflow command and passes all call data to the receiving contact center.

How the TS Interacts with the Workflow Server

The Workflow Server is a server that can be customized to control call flow logic. It is used primarily for call qualification, data pre-fetch, specialized routing, notifying a client application of a call's arrival, and other tasks performed before a call is routed to a customer service representative or other personnel.

Typically, the Legacy Aspect TS sends an event (TS.SendData) to the Workflow Server when an Avaya Computer Telephony-enabled CCT is run for an incoming call. The call has arrived and the Aspect has not yet completed the CCT; it is waiting for the Legacy Aspect TS to provide routing information. When the Workflow Server receives this event, it performs those actions have been programmed, such as ANI lookup or data pre-fetch. Customarily, the last step is a method invocation (TS.ReceiveData) on the Legacy Aspect TS to route the call to the next destination in the call flow.

Channels and Channel Ranges

A separate Workflow session, or channel, is required for each call that comes in on a given trunk. The Workflow server must be configured to provide enough simultaneous channels (a channel range) to handle the maximum number of calls that are likely to come in on that trunk.

To set up a channel range for use by the Legacy Aspect TS:

1. From the **Start** menu, choose **Programs>Avaya Interaction Center>IC Manager**. The **IC Manager** main window appears.
2. Select the **Server** tab at the **IC Manager** main window.

3 Select the **Workflow Server** to display the **Server Editor**.

4 Select the **Channels** tab at the **Server Editor** to display the **Channel Editor**.

5 In the **Channel Range** field, type a number that represents the maximum number of calls that are likely to come in on the trunk at one time.

6 In the **Service** field, select **TS**.

7 In the **Criteria** field, type *t and click the **Ok** button. The **Channel Editor** closes and the **Server Editor** regains focus.

8 In the **Server Editor**, select the line of text that describes the information you just specified and click the **Handshake** button. The **Channel Association** dialog is displayed. The **Channel Range** and **Service Interface** fields display the values you previously specified.

9 In the **Event** field, type the name of the event that is to trigger a flow (e.g., TS.IncomingCall).

10 In the **Flow** field, type the name of the flow that is to be triggered, in the format projectname.flowname and click the **Ok** button. The **Channel Association** dialog closes and the **Server Editor** regains focus.

11 In the **Server Editor**, click the **Ok** button to apply your changes and return to the **IC Manager** main window.

Channel ranges are configured in Workflow server via the Channels tab. For details, refer to *IC Administration Volume 1: Servers & Domains*.

CHAPTER 5
IDL AND METHOD DESCRIPTIONS

This chapter contains the Interface Definition Language (IDL) for the Legacy Aspect TS as it is implemented for Aspect CallCenter, a list of all methods, and a description of each method.

Background Information

All call-based commands (such as transfer, hold, etc.) have two method variations: one that includes the EDUID as an input parameter and one that does not. Using the EDUID as an input parameter is the recommended method of controlling all calls. It is particularly recommended for phones with more than one line. If the EDUID is not used, the Legacy Aspect TS does not know which phone line the command applies to so it applies the command to the next call in its stack.

This chapter describes only the method variations using the EDUID. The non-EDUID methods, which are supported, are listed in the Legacy Aspect TS IDL.

How a Client Joins an EDU's List of Interested Parties

The EDU Server maintains a list of clients interested in each EDU. When a client is no longer interested in the EDU, the client invokes the VDU.Terminate() method, which removes the client's name from the list. When the list is empty, the EDU is terminated.

A successful Transfer(), TransferInit(), or ConferenceInit() method invocation adds the third party (the party to whom the call is being transferred or who is being included in a conference) to the list of interested parties for the EDU. That client must eventually invoke a VDU.Terminate() method to remove the client's name from the list of interested parties. Another process may issue the Terminate on a client's behalf.

If the third party is not logged in at the time of the Transfer or Conference, the Legacy Aspect TS cancels the request and the client's name is not added to the list. However, if the client is logged in, regardless of whether or not the request is subsequently canceled, the list is updated.

A general rule to follow is that if the client receives a TS.IncomingCall.event, the client must eventually invoke a VDU.Terminate() method.

If a Terminate is not invoked, the EDU will eventually time out and be terminated automatically.

Interface Definition Language (IDL)

The following shows the Interface Definition Language (IDL) for the Legacy Aspect TS as it is implemented for Aspect CallCenter.

```
interface TS : General{

   ORBStatus Answer(void);
   ORBStatus AnswerVDU( in VDU_ID vduid );

   ORBStatus Assign( in string criteria );

   ORBStatus Busy( void );
   ORBStatus BusyTerminate( void );

   ORBStatus ConferenceCancel( void );
   ORBStatus ConferenceCancelVDU( in VDU_ID vduid );
   ORBStatus ConferenceComplete( void );
   ORBStatus ConferenceCompleteVDU( in VDU_ID vduid );
   ORBStatus ConferenceInit( in string dest );
   ORBStatus ConferenceInitVDU( in VDU_ID vduid, in string dest );

   oneway void Deassign( void ); (inherited)
   ORBStatus GetPhoneInfo( in string selection, out SeqCouple info )
   ORBStatus Hangup( void );
   ORBStatus HangupVDU( in VDU_ID vduid );

   ORBStatus Hold( void );
   ORBStatus HoldVDU( in VDU_ID vduid );
   ORBStatus HoldReconnect( void );
   ORBStatus HoldReconnectVDU( in VDU_ID vduid );

   ORBStatus Login(in string login, in string password,
                   in string queue, in string ext );
   ORBStatus Logout( in string queue, in string ext );

   ORBStatus MakeCall( in string dest );
   ORBStatus MakeCallVDU( in string dest, out VDU_ID vduid );
   ORBStatus MakeCallSetVDU( in VDU_ID vduid, in string dest );

   ORBStatus Ready( void );
   ORBStatus ReceiveData( in SeqCouples values );

   ORBStatus SwapHeld();

   ORBStatus Transfer( in string dest );
   ORBStatus TransferVDU( in VDU_ID vduid, in string dest );
   ORBStatus TransferCancel( void );
   ORBStatus TransferCancelVDU( in VDU_ID vduid );
   ORBStatus TransferInit( in string dest );
   ORBStatus TransferInitVDU( in VDU_ID vduid, string dest );
   ORBStatus TransferComplete( void );
   ORBStatus TransferCompleteVDU( in VDU_ID vduid );
   ORBStatus WrapUp( void );
};
```

List of Generic TS Methods

This section lists the generic methods available for use with the Legacy Aspect TS. In the list, the first party is the client invoking the method, the second party is the party to whom the first party is connected, and the third party is the party brought into an existing call.

Method	Description
AnswerVDU	Answer the phone
Assign	Create a session with the TS
Busy	Make an ACD teleset unavailable to receive ACD calls
BusyTerminate	Make an ACD teleset available to receive ACD calls
ConferenceCancelVDU	Cancel conference, hangup third party, retrieve second party from hold
ConferenceCompleteVDU	Retrieve the second party from hold and join the conference
ConferenceInitVDU	Put the second party on hold and call a third party
Deassign	Terminate a session
GetPhoneInfo	Return switch-dependent information about the state and mode of the phone
HangupVDU	Hang up phone
HoldReconnectVDU	Retrieve the party from hold
HoldVDU	Hold party
Login	Login ACD phone
Logout	Logout ACD phone
MakeCall	Initiate a call
MakeCallSetVDU	Initiate a call using an existing EDU, and set new values in the EDU
MakeCallVDU	Initiate a call using an existing EDU
Ready	Make an ACD teleset available to receive ACD calls
ReceiveData	Used by the Workflow server to respond to an Incoming Call event or a SendData event
SwapHeld	Swap states of busy and active lines
TransferCancelVDU	Cancel transfer, hang up third party, retrieve second party from hold
TransferCompleteVDU	Hang up the party initiating the transfer
TransferInitVDU	Put the second party phone on hold and call a third party
TransferVDU	Transfer, momentarily putting the second party on hold (blind transfer)
WrapUp	Have an ACD agent enter wrap-up state

Descriptions of Generic TS Methods

TS.AnswerVDU

IDL Syntax `ORBStatus AnswerVDU( in VDU_ID vdu_id );`

Description Answer a call, changing the call's state from "alerting" to "answered." Invoked in response to an IncomingCall event.

Input Parameters

Value	Description
vdu_id	EDUID of the call.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_PARAMETER	EDUID is invalid.
VESP_FAILURE	Request has failed; possible internal protocol problems.
VESP_BAD_SESSION	Session is invalid.
VESP_RESOURCE_NOT_AVAILABLE	No call to answer.
VESP_SERVICE_NOT_AVAILABLE	Switch does not have this function.

Example

```
VDU_ID vdu_id = "3016ace000700007800002c1b580002";

Environment ev;

status = Vesp_Request( "TS.AnswerVDU", callback, user_data, session, vdu_id );
```

TS.Assign

IDL Syntax `ORBStatus Assign( in string criteria );`

Description Create a session with the TS. Once a session is created, events corresponding to the monitoring criteria will be sent to the client.

Input Parameters

Input parameters correspond to criteria for monitoring and controlling devices during the current session.

Criterion	Description
number	An agent ID number (to monitor a phone station).
login ID	A login ID (to monitor an agent).
*t	The next available trunk. Used by the Workflow Server to monitor voice response lines.
*txxxx	A specific trunk, typically one connected to an IVR or some other off-switch equipment.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_ASSIGN_FAILURE	Assign has failed.
VESP_BAD_SESSION	Client ID is invalid.

Example

```
status = Vesp_Assign_Request( "TS.Assign", &ev, callback, user_data,
event_callback, session, "5112" );
```

TS.Busy

IDL Syntax `ORBStatus Busy( void );`

Description Make an ACD teleset unavailable for ACD calls. No calls will be received until a BusyTerminate or Ready is received. ACD calls will be blocked; direct calls will not be blocked.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_FAILURE	Request has failed; possible internal protocol problems.
VESP_SERVICE_NOT_AVAILABLE	Service not available.

Example

```
status = Vesp_Request( "TS.Busy", callback, user_data, session );
```

TS.BusyTerminate

IDL Syntax `ORBStatus BusyTerminate( void );`

Description Make an ACD phone available to receive calls. The phone would be in the Ready state.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_FAILURE	Request has failed; possible internal protocol problems.
VESP_SERVICE_NOT_AVAILABLE	Service not available.

Example

```
status = Vesp_Request( "TS.BusyTerminate", callback, user_data,
session );
```

TS.ConferenceCancelVDU

IDL Syntax `ORBStatus ConferenceCancelVDU( in string vdu_id );`

Description This function cancels a conference begun with the ConferenceInit function.

Input Parameters

Value	Description
vdu_id	EDUID of the call.

Returns

Value	Description
vesp_success	Request was successful.

Exceptions

Value	Description
vesp_bad_session	Session is invalid.
vesp_bad_vduid	EDUID is invalid.
vesp_failure	Request has failed; possible internal protocol problems.
vesp_illegal_state	State of the phone is not compatible with operation.

Example

```
VDU_ID vdu_id = "3016ace000700007800002c1b580002";

status = Vesp_Request( "TS.ConferenceCancelVDU", callback, user_data, session,
vdu_id );
```

TS.ConferenceCompleteVDU

IDL Syntax `ORBStatus ConferenceCompleteVDU( in string vdu_id );`

Description This function completes the conference initiated with the ConferenceInit function. The party on hold is joined to the other calls.

Input Parameters

Value	Description
vdu_id	EDUID of the call.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_PARAMETER	EDUID is invalid.
VESP_BAD_SESSION	Session is invalid.
VESP_BAD_VDUID	EDUID is invalid.
VESP_FAILURE	Request has failed; possible internal protocol problem.
VESP_RESOURCE_NOT_AVAILABLE	No call to conference.

Example

```
VDU_ID vdu_id = "3016ace000700007800002c1b580002";

status = Vesp_Request( "TS.ConferenceCompleteVDU", callback, user_data, session,
vdu_id );
```

TS.ConferenceInitVDU

IDL Syntax

```
ORBStatus ConferenceInitVDU( in string vdu_id, in string dest );
```

Description

This function places a party on hold and dials a third party. If this function fails, every effort is made to retrieve the party on hold automatically. The EDUID is passed in the incoming call event the end point receives.

If successful, this function places the third party on the EDU's list of interested parties. Refer to "How a Client Joins an EDU's List of Interested Parties," on page 39 for more information.

Input Parameters

Value	Description
dest	Destination to include in the conference. The destination can be any of the following: Agent ID number — The agent's teleset number is included in the conference. cctxxx — The call is transferred to the Call Control Table (CCT) specified by xxx. Name — The name is looked up in the Directory. If the telephone number field associated with the name contains an extension number, that extension will be included in the conference. If the telephone number field contains "cctxxx" (where xxx is a Call Control Table number), the call is transferred to the CCT specified by xxx.
vdu_id	EDUID of the call.
ddxxxx	Dial digits; a specific number to be dialed, typically used to place an outgoing call or to place an internal call to another agent (even if that agent's phone set is "busy" from the perspective of the switch).

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_BAD_VDUID	EDUID is invalid.
VESP_BUSY	Destination was busy.
VESP_FAILURE	Request has failed; possible internal protocol problem.

Example

```
VDU_ID vdu_id = "3016ace000700007800002c1b580002";

status = Vesp_Request( "TS.ConferenceInitVDU", callback, user_data, session,
vdu_id );
```

TS.Deassign

IDL Syntax `oneway void Deassign( void );`

Description Terminate a session with the TS. Once a session is terminated, the flow of events from the TS to the client will cease.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.

Example

```
status = Vesp_Deassign_Request( "TS.Deassign", &ev, NULL, OUL,
session );
```

TS.GetPhoneInfo

IDL Syntax `ORBStatus GetPhoneInfo( in string phonenumber, out SeqCouple values );`

Description Returns switch-dependent information about the state and mode of the phone.

Input Parameters

Value	Description
phone number	Currently must be blank; only the default phone is valid.

Output Parameters A sequence of the following three couples: state, aspect_status, and mode.

Value	Possible Parameters
state	active, idle, null
aspect_status	V (vacant), I (idle) A, (available), N (busy/inbound), O (busy/outbound), S (busy/inside), or X (busy/other)
mode	busy, ready, notready

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_FAILURE	Request failed.

Example

```
status = Vesp_Request( "TS.GetPhoneInfo", callback, user_data, session, "",
values );
```

TS.HangupVDU

IDL Syntax

```
ORBStatus HangupVDU( in string vdu_id );
```

Description

This function hangs up the voice portion of a call. The EDU remains active for any call wrap-up activities that are desired by the application.

Note: To place an agent into a wrap-up state after hangup, the switch must be programmed to do so.

Input Parameters

Value	Description
vdu_id	EDUID of the call.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_BAD_VDUID	EDUID is invalid.
VESP_FAILURE	Request has failed; possible internal protocol problem.
VESP_SERVICE_NOT_AVAILABLE	Service is not available.

Example

```
VDU_ID vdu_id = "3016ace000700007800002c1b580002";
status = Vesp_Request( "TS.HangupVDU", callback, user_data, session, vdu_id );
```

TS.HoldReconnectVDU

IDL Syntax `ORBStatus HoldReconnectVDU( in string vdu_id );`

Description Takes a phone off hold and ensures that there are no calls currently active.

Input Parameters

Value	Description
vdu_id	EDUID of the call.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_BAD_VDUID	EDUID is invalid.
VESP_FAILURE	Request has failed; possible internal protocol problem.
VESP_RESOURCE_NOT_AVAILABLE	There is no call to take off hold.

Example

```
VDU_ID vdu_id = "3016ace000700007800002c1b580002";

status = Vesp_Request( "TS.HoldReconnectVDU", callback, user_data, session,
vdu_id );
```

TS.HoldVDU

IDL Syntax `ORBStatus HoldVDU( in string vdu_id );`

Description This function places the voice portion of the call on hold. The EDU may still be acted on by the application.

Input Parameters

Value	Description
vdu_id	EDUID of the call.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_BAD_VDUID	EDUID is invalid.
VESP_SERVICE_NOT_AVAILABLE	Service is not available.
VESP_RESOURCE_NOT_AVAILABLE	There is no call to put on hold.

Example

```
VDU_ID vdu_id = "3016ace000700007800002c1b580002";
status = Vesp_Request( "TS.HoldHoldVDU", callback, user_data, session, vdu_id );
```

TS.Login

IDL Syntax

```
ORBStatus Login( in string login, in string password, in string queue, in string ext );
```

Description

The Login function logs in the teleset.

Input Parameters

Value	Description
login	Logical identification of the agent (i.e., the Agent ID).
password	Phone password, for switch.
queue	Not implemented.
ext	Equipment number.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_FAILURE	Request failed.

Example

```
status = Vesp_Request( "TS.Login", callback, user_data, session, "vesp", "", "",
"4009" );
```

TS.Logout

IDL Syntax

```
ORBStatus Logout( in string queue, in string ext );
```

Description This function logs off the teleset from the switch, if the teleset is in a state in which it can be logged off.

Input Parameters

Value	Description
queue	Ignored. Included for compatibility with other versions of the Legacy Aspect TS.
ext	Ignored. Included for compatibility with other versions of the Legacy Aspect TS.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_FAILURE	Request failed.

Example

```
status = Vesp_Request( "TS.Logout", callback, user_data, session,
 "", "" );
```

TS.MakeCall
TS.MakeCallSetVDU
TS.MakeCallVDU

IDL Syntax

```
ORBStatus MakeCall( in string dest );

ORBStatus MakeCallSetVDU( in string vdu_id, in string dest );

ORBStatus MakeCallVDU( in string dest, out string vdu_id );
```

Description

These functions initiate a call attempt. MakeCall generates a new EDU but does not return the EDU. MakeCallSetVDU uses an existing, specified EDU. MakeCallVDU generates a new EDU and returns it.

The status of the call attempt is reported back to the client.

The MakeCall functions fail if it receives any type of busy event, recording the call attempt details and reason for termination.

It is possible that a Busy event will be generated if the destination is busy.

A Connect event is generated if control is successfully handed off to the switch. (A connect event does not necessarily indicate call completion.)

A disconnect event is *not* generated if the phone is hung up before a connect event can be generated.

Input Parameters

Value	Description
dest	Destination to include in the conference. The destination can be any of the following: Agent ID number — The call is placed to the agent's teleset. cctxxx — The call is placed to the Call Control Table (CCT) specified by xxx. Name — The name is looked up in the Directory. If the telephone number field associated with the name contains an extension number, the call is placed to that extension. If the telephone number field contains "cctxxx" (where xxx is a Call Control Table number), the call is placed to the CCT specified by xxx.
ddxxxx	Dial digits; a specific number to be dialed, typically used to place an outgoing call or to place an internal call to another agent (even if that agent’s phone set is "busy" from the perspective of the switch).
vdu_id	EDUID of the call.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_BAD_VDUID	EDUID is invalid.
VESP_BUSY	Destination was busy.
VESP_FAILURE	Request has failed; possible internal protocol problem.

Example

```
VDU_ID vdu_id = "3016ace000700007800002c1b580002";

status = Vesp_Request( "TS.MakeCallSetVDU", callback, user_data,  session,
"400"’ &vdu_id );

VDU_ID vdu_id;

status = Vesp_Request( "TS.MakeCallSetVDU", &ev, callback, user_data,  session,
"400"’ &vdu_id );
```

TS.Ready

IDL Syntax

```
ORBStatus Ready( void );
```

Description

Places a telephone set in the ready state in preparation for receiving a telephone call.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_FAILURE	Request has failed; possible internal protocol problem.
VESP_SERVICE_NOT_AVAILABLE	Service is not available.

Example

```
status = Vesp_Request( "TS.Ready", callback, user_data, session );
```

TS.ReceiveData

IDL Syntax `ORBStatus ReceiveData(("data_a", "value_a") ("data_b", "value_b") ("data_c", "value_c") ("data_d", "value_d") ("cct", "value_cct") ("resp", "value_resp"));`

Description Invoked by WorkfFlow server to respond to a SendData event or an IncomingCall event.

Input Parameters

Input parameters are the data to be passed to the Aspect switch. Possible values are `"data_a", "data_b", "data_c", "data_d", "cct", "resp"`.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_FAILURE	Request has failed; possible internal protocol problem.
VESP_SERVICE_NOT_AVAILABLE	Service is not available.

Example

```
status = Vesp_Request( "TS.ReceiveData", callback, user_data, session );
```

TS.SwapHeld

IDL Syntax `ORBStatus SwapHeld();`

Description If one outside line on a teleset is on hold and another line is active, invoking this method will swap their states.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_FAILURE	Request has failed; possible internal protocol problem.
VESP_SERVICE_NOT_AVAILABLE	Service is not available.

TS.TransferCancelVDU

IDL Syntax `ORBStatus TransferCancelVDU( in string vdu_id );`

Description This function cancels a transfer begun with the TransferInit function.

Input Parameters

Value	Description
vdu_id	EDUID of the call.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_FAILURE	Request has failed.
VESP_ILLEGAL_STATE	State of phone is not compatible with operation.

Example

```
VDU_ID vdu_id = "3016ace000700007800002c1b580002";

status = Vesp_Request( "TS.TransferCancelVDU", callback, user_data, session,
vdu_id );
```

TS.TransferCompleteVDU

IDL Syntax `ORBStatus TransferCompleteVDU( in string vdu_id );`

Description This function completes the transfer started with the TransferInit function. The party on hold is connected with the called third party and the first party (the originator of the transfer) is hung up.

Input Parameters

Value	Description
vdu_id	EDUID of the call.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_FAILURE	Request has failed.
VESP_ILLEGAL_STATE	State of phone is not compatible with operation.

Example

```
VDU_ID vdu_id = "3016ace000700007800002c1b580002";

status = Vesp_Request( "TS.TransferCompleteVDU", callback, user_data,
session, vdu_id );
```

TS.TransferInitVDU

IDL Syntax `ORBStatus TransferInitVDU( in string vdu_id, in string dest );`

Description This function places the call initiator on hold and calls a third party. If this function fails, the call initiator is retrieved from hold. The EDUID is passed in the incoming call event the end point receives.

If successful, this function places the third party on the EDU's list of interested parties. Refer to "How a Client Joins an EDU's List of Interested Parties," on page 39 for more information.

Input Parameters

Value	Description
dest	The destination can be any of the following: Agent ID number — The call is transferred to the agent's teleset. cctxxx — The call is transferred to the Call Control Table (CCT) specified by xxx. Name — The name is looked up in the Directory. If the telephone number field associated with the name contains an extension number, the call is transferred to that extension. If the telephone number field contains "cctxxx" (where xxx is a Call Control Table number), the call is transferred to the CCT specified by xxx.
ddxxxx	Dial digits; a specific number to be dialed, typically used to place an outgoing call or to place an internal call to another agent (even if that agent's phone set is "busy" from the perspective of the switch).
vdu_id	EDUID of the call.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_ILLEGAL_STATE	State of phone is not compatible with operation.
VESP_BUSY	Destination was busy.
VESP_FAILURE	Request has failed; possible internal protocol problem.

Example

```
VDU_ID vdu_id = "3016ace000700007800002c1b580002";

status = Vesp_Request( "TS.TransferInitVDU", callback, user_data, session,
vdu_id, "5000" );
```

TS.TransferVDU

IDL Syntax `ORBStatus TransferVDU( in string vdu_id, in string dest );`

Description This function transfers a call and its EDU to a destination. The second party (the party being transferred) is momentarily placed on hold, and the third party (the party receiving the transfer) receives an IncomingCall event.

If successful, this function places the third party on the EDU's list of interested parties. Refer to "How a Client Joins an EDU's List of Interested Parties," on page 39 for more information.

Input Parameters

Value	Description
dest	The destination can be any of the following: Agent ID number — The call is transferred to the agent's teleset. cctxxx — The call is transferred to the Call Control Table (CCT) specified by xxx. Name — The name is looked up in the Avaya IC Directory. If the telephone number field associated with the name contains an extension number, the call is transferred to that extension. If the telephone number field contains "cctxxx" (where xxx is a Call Control Table number), the call is transferred to the CCT specified by xxx.
ddxxxx	Dial digits; a specific number to be dialed, typically used to place an outgoing call or to place an internal call to another agent (even if that agent's phone set is "busy" from the perspective of the switch).
vdu_id	EDUID of the call.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_ILLEGAL_STATE	State of phone is not compatible with operation.
VESP_BUSY	Destination was busy.
VESP_FAILURE	Request has failed; possible internal protocol problem.
resource_not_available	The resource is currently unavailable.

Example

```
VDU_ID vdu_id = "3016ace000700007800002c1b580002";

status = Vesp_Request( "TS.TransferVDU", callback, user_data, session, vdu_id,
"4500" );
```

TS.WrapUp

IDL Syntax

```
ORBStatus WrapUp( void );
```

Description This function places the phone in a wrap-up state.

Returns

Value	Description
VESP_SUCCESS	Request was successful.

Exceptions

Value	Description
VESP_BAD_SESSION	Session is invalid.
VESP_FAILURE	Request has failed; possible internal protocol problem.
VESP_SERVICE_NOT_AVAILABLE	Service is not available.

Example

```
status = Vesp_Request ( "TS.WrapUp", callback, user_data, session );
```

CHAPTER 6
EVENTS

This chapter describes the Legacy Aspect TS events.

List of Events

The following events are generated by the Legacy Aspect TS

Event	Description
Busy	Call is busy at destination.
Connect	Call has been connected with end point,
Disconnect	Call has been terminated.
Drop	Party has been dropped from the call.
IncomingCall	A call is present at the device.
Ring	The TS has received a Call Offered event from the Aspect or a MakeCall request.
SendData	The TS has received a Call Information Message.
ServerFailed	The server has failed.

The events listed in the following table are supported only for Aspect Release 6.*x* or 7.2 (Application Bridge version 4) with the Event Bridge option

Event	Description
AuxWork	Phone set is not ready to receive incoming calls.
Hold	Call has been put on hold.
HoldReconnect	Call has been taken off hold.
Login	Agent has logged on the phone set.
Logout	Agent has logged out of the phone set.

Event	Description
Ready	Agent has become available to take incoming calls.
Reserved	Agent has gone to the "06" state (the definition of this state is reserved by Aspect).
WrapUp	Agent's teleset has gone to the wrapup state.

Note: This chapter lists some of the fields returned with each event. Additional information may be returned. The fields are not necessarily listed in the order in which they are returned.

Event Descriptions

TS.AuxWork

This event informs the client that a phone set has gone to the idle state (it is not ready to receive incoming calls). This event is supported only for Aspect Release 6.*x* or 7.2 (Application Bridge version 4) with the Event Bridge option.

TS.Busy

This event informs the client application that the call it is attempting to make received a busy signal. The client is not connected to the recipient.

Returns

Value	Description
vdu_id	EDUID of this call.
call_ref_id	Call reference ID of call, used for debugging purposes only.

TS.Connect

This event informs the client application that a call has been connected. A Connect event results from a MakeCall, TransferInit, or ConferenceInit invocation.

Returns

Value	Description
vdu_id	EDUID of this call.
call_ref_id	Call reference ID of call, used for debugging purposes only.

TS.Disconnect

This event informs the client application that a call has been disconnected.

Returns

Value	Description
vdu_id	EDUID of this call.
call_ref_id	Call reference ID of call, used for debugging purposes only.

TS.Drop

This event informs the client application that one party (in a call made up of more than two parties) has been dropped from a call. The call itself remains active.

Returns

Value	Description
vdu_id	EDUID of this call.
call_ref_id	Call reference id of call, used for debugging purposes only.
dest	The actual phone number to which the call was placed by the switch. This is typically an equipment (phone) number.

TS.Hold

This event informs the client application that a call has been put on hold. This event is supported only for Aspect Release 6.*x* or 7.2 (Application Bridge version 4) with the Event Bridge option.

Returns

Value	Description
vdu_id	EDUID of this call.
call_ref_id	Call reference id of call, used for debugging purposes only.
dest	The actual phone number to which the call was placed by the switch. This is typically an equipment (phone) number.

TS.HoldReconnect

This event informs the client application that a call has been retrieved from hold. This event is supported only for Aspect Release 6.*x* or 7.2 (Application Bridge version 4) with the Event Bridge option.

Returns

Value	Description
vdu_id	EDUID of this call.
call_ref_id	Call reference id of call, used for debugging purposes only.
dest	The actual phone number to which the call was placed by the switch. This is typically an equipment (phone) number.

TS.IncomingCall

This event informs the client application that a telephone line to which it is assigned is in the ringing state or has received a call set-up request. In addition to telling the client that its associated telephone number is alerting, the event passes the EDUID of the call to the client.

Once the client has received an IncomingCall.event, that client must invoke a VDU.Terminate method when it is no longer interested in the EDU.

Returns

Value	Description
vdu_id	EDUID of this call.
ani	Automatic Number Identification; the caller's 10-digit telephone number.
dnis	Dialed Number Identification Service; the number dialed.
call_ref_id	Call reference ID of call, assigned by the switch. Used for debugging purposes only.
orig	The equipment number (physical phone number) of the party that placed the call or initiated the transfer or conference to the party receiving this event.
dest	The actual phone number to which the call was placed by the switch. This is typically an equipment (phone) number.
called	For an outbound call, the number called as assigned by the client application through the Transfer, Conference, or MakeCall methods. At first this is equivalent to the DNIS, but will change as the call is routed through the system.

TS.Login

This event informs the client that the agent has logged onto the phone set. This event is supported only for Aspect Release 6.*x* or 7.2 (Application Bridge version 4) with the Event Bridge option.

TS.Logout

This event informs the client that the agent has logged off of the phoneset. This event is supported only for Aspect Release 6.*x* or 7.2 (Application Bridge version 4) with the Event Bridge option.

TS.Ready

This event informs the client that the agent has become available to take incoming calls. This event is supported only for Aspect Release 6.*x* or 7.2 (Application Bridge version 4) with the Event Bridge option.

TS.Reserved

This event informs the client that the Agent has gone to the "06" state. (The definition of this state is reserved by Aspect.) This event is supported only for Aspect Release 6.*x* or 7.2 (Application Bridge version 4) with the Event Bridge option.

TS.Ring

This event informs the client that the TS has received a Call Offered event from the Aspect in response to a MakeCall or Transfer request.

Returns

Value	Description
vdu_id	EDUID of this call.
ani	Automatic Number Identification; the caller's 10-digit telephone number.
dnis	Dialed Number Identification Service; the number dialed.
call_ref_id	Call reference ID of call, assigned by the switch. Used for debugging purposes only.
type	The call type, such as new or txfer.
Data_c, d, e	The data fields.

TS.SendData

This event informs the client assigned with *t of the arrival of a Call Information Message (CIM).

This event is not sent in response to CIMs with a subtype of newtrans, startcall, gv*, sv* (with the exception of svlanguage), ev*, or ed*.

Returns

Value	Description
vdu_id	EDUID of this call.
call_ref_id	Call reference ID of call, assigned by the switch. Used for debugging purposes only.
subtype	Subtype of the CIM.
Data A - E	The data fields.

TS.ServerFailed

This event informs the client application that the server has failed.

TS.WrapUp

This event informs the client that the agent's teleset has gone to the wrapup state. This event is supported only for Aspect Release 6.*x* or 7.2 (Application Bridge version 4) with the Event Bridge option.

CHAPTER 7
ALARMS

This chapter lists and describes the alarms that can be generated by the Legacy Aspect TS.

List of Alarms

Alarm Name	Priority	Description	Cause/Recommended Action
CSRProblem	high	General Aspect agent problems.	Typical causes might be the Legacy Aspect TS cannot find the trunk on connect, not enough Workflow server sessions are logged in, the Workflow server is disabled, or the agent connected is not logged in to Avaya IC and assigned to the Legacy Aspect TS.
DirLoadFailure	high	Could not load local tables from Directory Server.	The Directory Server has failed. Check the status of the Directory Server.
DirUpDateFail	high	Could not update local tables from Directory Server.	The Directory Server has failed. Check the status of the Directory Server.
InvalidInit	low high emergency	Text describing the invalid field	Indicates missing or invalid information in the server configuration. Emergency alarms of this type will prevent the TS from starting.
LinkConnect	info	Link to the switch has been established.	The TS started successfully or a lost link was reestablished. No action required.
LinkConnectFail	high	Initial connection or re-connection to the switch has failed.	The TS could not talk to the switch. Check the TS configuration in IC Manager and the network configuration on the network host machine. Ensure that the cable is not unplugged.
LinkLost	high	Lost a connection to the switch.	If there is more than one link to the switch, the TS may continue to operate. Periodically the connection will try to re-establish.
LinkLostAll	Emergency	Lost all connections to the switch.	This is a catastrophic failure and disconnect events will be sent to the client application.

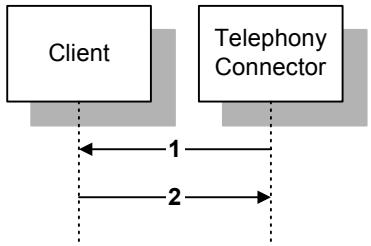
Alarm Name	Priority	Description	Cause/Recommended Action
LinkNotDefined	high	Link to the switch is not defined. Server will exit.	The Link parameter in the TS configuration must be defined.
Problem	high, low	General TS problem. The problem is described in detail.	Depends on the problem reported.
VDUNotCreated	high	EDU could not be created for call.	The EDU Server is not responding quickly enough to TS requests. Check the status of the EDU Server.

CHAPTER 8
CALL FLOWS

This chapter illustrates and describes some generic call flows.

Route Call

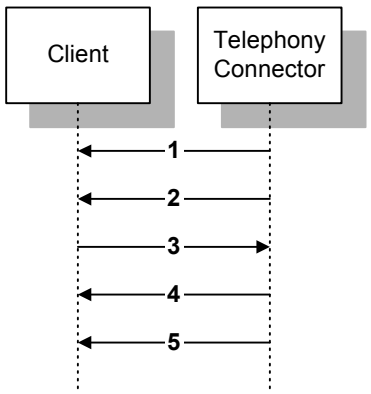
The following diagram illustrates the events and methods that are exchanged when the Legacy Aspect TS routes a call to a client.



Step	Description
1	The TS sends TS.IncomingCall.event(*vdu_id, call_id, ani, dnis, ctype*) to client.
2	Client invokes TS.ReceiveData(*values*) method on the TS.

Inbound Call

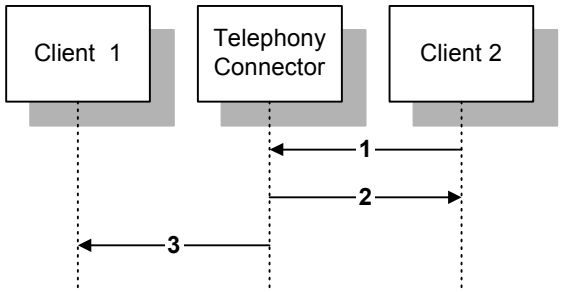
The following diagram illustrates the methods and events that are exchanged after a call arrives at the Legacy Aspect TS from the PBX or ACD. The sequence ends with the call being terminated by the client.



Step	Description
1	The TS sends a TS.Incoming.event(*vdu_id, ani, dnis, call_ref_id, ctype*) to the client.
2	The TS sends a TS.Connect.event(*vdu_id, call_ref_id*) to the client.
3	Client invokes the TS.HangupVDU(*vdu_id*) method on the TS.
4	The TS returns a TS.HangupVDU.response(*vdu_id*) to the client.
5	The TS sends a TS.Disconnect.event(*vdu_id, call_ref_id*) to the client.

Outbound Call

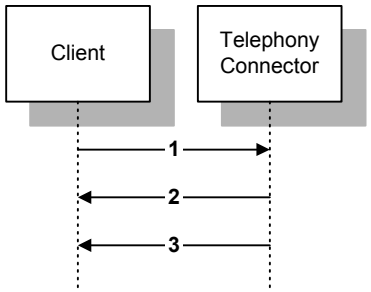
The following diagram illustrates the methods and events that are exchanged during an outbound call. In this example, Client 1 is either an agent or a queue and Client 2 is an automatic dialer.



Step	Description
1	Client 2 invokes TS.MakeCallVDU(*dest*) method on the TS.
2	The TS returns TS.MakeCallVDU.response(*dest.vdu_id*) to Client 2.
3	The TS sends TS.Connect.event(*vdu_id, call_ref_id*) to Client 1.

Busy Destination

The following diagram illustrates the methods and events that are exchanged when the call destination is busy.



Step	Description
1	Client invokes TS.MakeCallVDU(*dest*) on the TS.
2	The TS returns TS.MakeCallVDU.response(*dest.vdu_id*) to the client.
3	The TS also returns TS.Busy.event(*vdu_id.call_ref_id*) to the client.

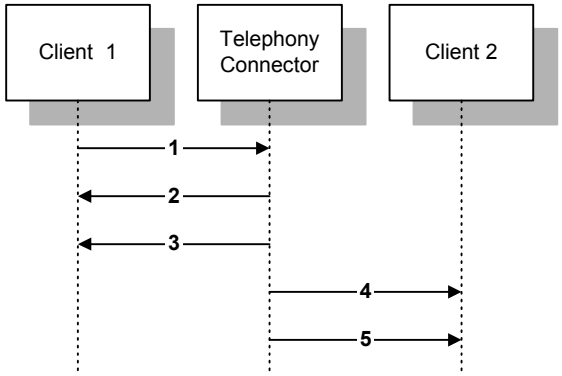
Note: A Busy event is generated when the destination is busy. A Busy exception is raised when the originator is busy.

Blind Transfer

The following diagram illustrates the methods and events that are exchanged during a blind transfer (a transfer in which the first agent hangs up before the next agent picks up the call).

ACD sets must be in the ready state. Calls will be answered automatically if the switch is configured to AutoAnswer or if AutoAnswer is set through VTel.

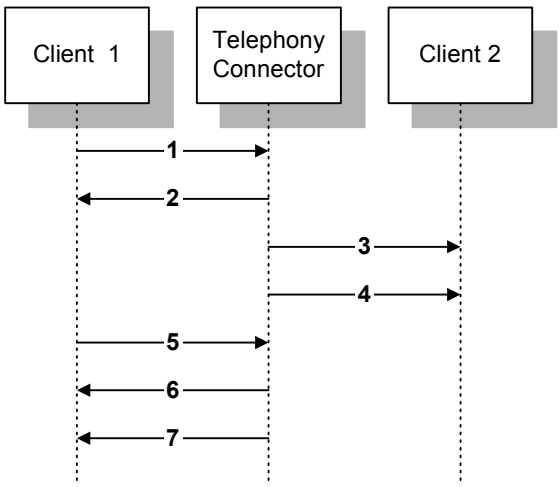
If the call originates within the switch system, additional events will be generated that are not indicated in this model.



Step	Description
1	Client 1 invokes TS.TransferVDU(*vdu_id, dest*) method on the TS.
2	The TS returns TS.TransferVDU.response(*dest, vdu_id*) to Client 1.
3	The TS also returns TS.Disconnect.event(*vdu_id, call_ref_id*) to Client 1.
4	The TS sends TS.IncomingCall.event (*vdu_id, call_ref_id, ani, dnis, ctype*) to Client 2.
5	The TS sends TS.Connect.event(*vdu_id, call_ref_id*) to Client 2.

Consultative Transfer

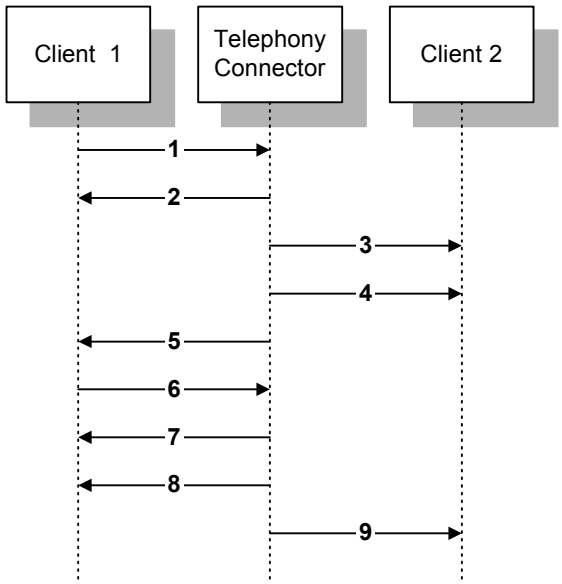
The following diagram illustrates the methods and events that are exchanged during a consultative transfer (a transfer in which the first agent remains on the line until the next agent picks up the call).



Step	Description
1	Client 1 invokes TS.TransferInitVDU(*vdu_id, dest*) method on the TS.
2	The TS returns TS.TransferInitVDU.response(*dest, vdu_id*) to Client 1.
3	The TS sends TS.IncomingCall.event(*vdu_id, call_ref_id, ani, dnis, ctype*) to Client 2.
4	The TS sends TS.Connect.event(*vdu_id, call_ref_id*) to Client 2.
5	Client 1 invokes TS.TransferCompleteVDU (*vdu_id*) method on the TS.
6	The TS returns TS.TransferCompleteVDU.response(*vdu_id*) to Client 1.
7	The TS sends TS.Disconnect.event(*vdu_id, call_ref_id*) to Client 1.

Internal Call

The following diagram illustrates the methods and events that are exchanged when a client places an internal call to another client. Note that an external call would have the same flow, except steps 3, 4, and 9 would not apply.



Step	Description
1	Client 1 invokes TS.MakeCallVDU(*dest*) method on the TS.
2	The TS returns TS.MakeCallVDU.response(*dest, vdu_id*) to Client 1.
3	The TS sends TS.IncomingCall.event(*vdu_id, ani, dnis, call_ref_id, ctype*) to Client 2.
4	The TS sends TS.Connect.event(*vdu_id, call_ref_id*) to Client 2.
5	The TS sends TS.Connect.event(*vdu_id, call_ref_id*) to Client 1.
6	Client 1 invokes TS.HangupVDU(*vdu_id*) method on the TS.
7	The TS sends TS.HangupVDU.response(*vdu_id*) to Client 1.
8	The TS sends TS.Disconnect.event(*vdu_id, call_ref_id*) to Client 1.
9	The TS sends TS.Disconnect.event(*vdu_id, call_ref_id*) to Client 2.

APPENDIX A
SAMPLE CALL CONTROL TABLES

This appendix contains a sample Call Control Table (CCT) that would be appropriate for handling a MakeCall, and a sample CCT that would be appropriate for handling a Transfer.

Refer to "Tracking Call and Agent Data," on page 17 for an overview of the use of CCTs and for a sample CCT for incoming calls.

The CCTs in this manual are for example only; in a production environment, users would substitute appropriate commands and values.

Default MakeCall CCT

When an agent places an outbound call, the Legacy Aspect TS builds a Place Call Request, which is sent over the application bridge to the Aspect CallCenter. The Place Call Request specifies a CCT to which control of the call is passed. The CCT used by the Place Call Request can be established in one of two ways:

- The CCT number can be passed as a parameter in the TS.MakeCall() function.
- The default MakeCall CCT, specified by the Legacy Aspect TS's makecallcct configuration parameter, can be used.

The following is an example of a very basic CCT used to connect inside callers with an outside line.

```
Step #     Command              -----------Attributes----------
1          EXTENSION            NUMBER > ____ or FROM VARIABLE  [$]
                                ON SUCCESS, GO TO STEP # 4
                                ON NO ANS, _________________
                                ON BUSY, ___________________
                                ON VARIANT, ________________
                                ON UNDEFINED, ______________
2          QUEUE                20 SECONDS
3          ANNOUNCEMENT         NUMBER > 3
4          DELAY                NUMBER OF SECONDS: 3
5          DISCONNECT
6          DIAL                 LAST__DIGITS or # FROM [$] or NUMBER:____
7          SEND CONNECT         LINK # 14  SUBTYPE ________ VAR LIST A-E
                                ON ERROR, __________________
8          CONNECT              ON NO ANSWER, EXECUTE __#__
```

Step 6 illustrates the DIAL command. The FROM field is used to specify a variable that contains the number to be dialed ($ for the dialed digits buffer).

Default Transfer CCT

For Avaya Computer Telephony to maintain accurate information about the status of a call, a CCT must be used to transfer the call rather than transferring the call with the Aspect's Inside Line feature.

The CCT used to handle a transferred call can be established in one of two ways:

- The CCT number can be passed as a parameter in the TS.TransferVDU() function.
- The default Transfer CCT, specified by the Legacy Aspect TS's txfercct configuration parameter, can be used.

The following is an example of a very basic CCT used to transfer a call.

```
Step #     Command              ------------Attributes----------
1          EXTENSION            NUMBER > ____ or FROM VARIABLE [D]
                                ON SUCCESS, GO TO STEP # 4
2          QUEUE                60 SECONDS
3          GOTO                 STEP NUMBER:__2
4          SEND CONNECT         LINK #> _11 SUBTYPE startcall VAR LIST A-E
5          CONNECT              ON NO ANSWER, EXECUTE STEP # __6
6          ANNOUNCEMENT         NUMBER > 72
7          ANNOUNCEMENT         NUMBER > 73
8          ANNOUNCEMENT         NUMBER > 74
9          DISCONNECT
```

Step 1 illustrates the EXTENSION command. For a blind transfer, the D variable contains the dialed digits, which is the number to which the call is transferred.

Step 4 illustrates the SEND CONNECT command, which specifies the link to be used and identifies the subtype.

APPENDIX B
SEND DATA COMMAND SUBTYPES

The Subtype field of the Send Data command is used to provide additional information about the nature of the data being sent to Avaya IC.

Data associated with the subtype is either part of the subtype name or is sent in the data_a field.

- Data that is entered by a caller using a touch-tone phone is placed in data_a.
- Data that is not entered using touch-tones, such as information generated by the CCT, cannot be stored in a data field. The Aspect appends this type of information to the subtype name. For example, to notify the Workflow server that announcement number 6 was played, the Aspect would send subtype ann_006.

Some of the subtypes from the Aspect cause the Legacy Aspect TS to return data. The returned data is stored in data_a, data_b, data_c, and data_d. data_e contains the EDUID. The Resp register is used to indicate errors. Resp is "N" (non-acknowledgement) if an error occurred and "A" (acknowledgement) if the command was successful. Subtypes that do not return data cannot return errors.

Subtypes can be processed by the Workflow server or by the Legacy Aspect TS. The following subtypes are handled by the Legacy Aspect TS; all others are handled by the Workflow server:

ed*xxxxxxxxxx*
ev*xxxxxxxxxx*
gv*xxxxxxxxxx*
newtrans
startcall
sv*xxxxxxxxxx*

The subtypes sent by the Aspect are described briefly in the following table and are described in more detail in the following sections. The *xxx* after some of the subtype names indicates data associated with that subtype.

The subtype names refer to data transfers with respect to the Aspect Call Control Table, not with respect to the Legacy Aspect TS.

Subtype	Description	Data	Returns
aniquery	Indicates whether there was an ANI match	None	Returns in Resp an "A" for success or an "N" for failure
ann*xxx*	Reports which announcement was played	None except the *xxx* that is part of the subtype name	None
ed*xxxxxxxxxx*	Sends an event message with associated data elements data_a through data_e to the EDU Server	data_a through data_e	Errors in Resp
ev*xxxxxxxxxx*	Sends an event message without data elements to the EDU Server	None	Errors in Resp
getrouting	Requests call routing information from Workflow Server	None	Routing code in data_a, agent's phone number in data_b, errors in Resp
gv*xxxxxxxxxx*	Retrieves one value from the EDU	None	Retrieved value in data_a, errors in Resp
newtrans	Changes the call reference ID of an existing call	None	New call reference ID
reason*xxx*	Indicates the reason code(s) to be used for wrapping up the call	data_a (optional), *xxx* that is part of the subtype name (optional)	None
startcall	Indicates that a new call has begun	data_a contains the ANI, data_b contains the DNIS	Account # in data_a, errors in Resp
sv*xxxxxxxxxx*	Sets one value in the EDU	data_a	Errors in Resp

aniquery ANI Query

This function is serviced indirectly by the Workflow server. It returns useful information only if the call flow implemented by a customized Workflow server is designed to query the ANI (CQS) Server when an incoming call arrives. The Workflow server must set the "ani_hit" flag in the Workflow server data structure for each call to "1" if the query resulted in a unique match. If this flag is set, subsequent SEND DATA commands with a subtype of "aniquery" will return "A".

If the Workflow server is not customized in this fashion, the aniquery subtype responds with "N" in all cases.

Return Values Resp contains "A" if the caller was previously identified by ANI, "N" if the ANI does not match a record in the call qualification database or if a startcall was not previously performed.

Errors If an error occurs, an "N" is returned in the Resp field.

Example Determine if caller was previously identified by ANI. Branch on results.

Step	Command	Attributes
1	SEND DATA	LINK#> 1 SUBTYPE aniquery VAR LIST A-E ON ERROR, EXECUTE CCT #999
2	RECEIVE DATA	LINK#> 1 ON NAK, EXECUTE CCT # 123 ON ERROR, EXECUTE CCT #999

ann*xxx* Announcement Played

This function informs the Workflow server that announcement number *xxx* was played. It is serviced indirectly by the Workflow server. The Workflow server looks for the announcement number in the Directory Server "_annEnglish" table. A couple with a name of "ann" is placed in the EDU if a description of this announcement is found in the table.

For example, if a SEND DATA is executed with subtype of "ann023", and if the Directory Server table "_annEnglish" is found to contain the couple

{"_023", "Super Important Announcement"}

then the following data would be placed in the EDU:

{"ann", "023 Super Important Announcement"}

If the announcement number is not in the table, or if the table does not exist, no entry is made in the EDU.

The "_annEnglish" table can be created and modified through IC Manager. For more details, refer to *IC Administration Volume 1: Servers & Domains*.

Return Values This subtype does not return any information.

Example

Step	Command	Attributes
1	ANNOUNCEMENT	NUMBER: 11
2	SEND DATA	LINK#> 1 SUBTYPE ann011 ON ERROR, EXECUTE CCT #999

ed*xxxxxxxxxx* Event With Data

An EDU event is added to the EDU for this call. The event name is the portion of the subtype following "ed". The event data elements are a sequence of couples with names "data_a", "data_b", etc., and values specified by the contents of the corresponding data fields in the SEND DATA command.

Refer to the *Electronic Data Unit Server Programmer's Guide* for more information on the events stored in an EDU.

Return Values Resp contains "N" for failure and "A" for success.

Example Send an event message indicating that CCT 123 was invoked. data_a through data_e are stored in the event.

Step	Command	Attributes
1	SEND DATA	LINK#> 1 SUBTYPE edcct123 VAR LIST A-E ON ERROR, EXECUTE CCT #999
2	RECEIVE DATA	LINK#> 1 ON NAK, EXECUTE CCT # 123 ON ERROR, EXECUTE CCT #999

ev*xxxxxxxxxx* Event Without Data

An EDU event is added to the EDU for this call. The event name is the portion of the subtype following "ev". The event data elements are not stored.

Refer to the *Electronic Data Unit Server Programmer's Guide* for more information on the events stored in an EDU.

Returns Resp contains "N" for failure and "A" for success.

Example Send an event message indicating that CCT 123 was invoked. No data is stored in the event.

Step	Command	Attributes
1	SEND DATA	LINK#> 1 SUBTYPE evcct123 VAR LIST A-E ON ERROR, EXECUTE CCT #999
2	RECEIVE DATA	LINK#> 1 ON NAK, EXECUTE CCT # 123 ON ERROR, EXECUTE CCT #999

getrouting Get Call Routing Directions

This function is serviced indirectly by the Workflow server. The data elements in the EDU for this call are evaluated against a table of script rules stored in the Directory Server. The resulting string, which could be an extension, queue, agent name, or other data, is placed in data element A.

The script rules table ("_scriptrules") can be created and modified using IC Manager.

The script rules table contains name-value pairs. The name represents a name for the rule; the name must begin with "rule". The value lists a routing extension (or other data to be returned), followed by a colon (":") and a logical rule based on a data element expected in the EDU.

EDU values are evaluated against the rules table. The rules are evaluated in order, and data is returned for the first rule satisfied. If no rules are satisfied, an empty string is returned.

For example, assume the script rules table contains the following entries:

```
rule_special    CSR11:dnis==1234
rule45          1200:Account_value<3000
rule_VIP        999:last_name==Smith|priority>=3
```

In this example, if the name-value pair {"Account_value","2250"} is in the EDU, the string "1200" will be returned in data element A. If the name-value pair {"dnis","1234"} is also present, the string "CSR11" will be returned instead (rules are evaluated in the order they occur in the table). The third rule would succeed if the first two failed, and the EDU contained the pairs {"last_name","Smith"} and {"priority","5"}.

Refer to *IC Administration Volume 1: Servers & Domains* for more information on script rules.

Return Values data_a contains a string if a hit was made in the script rules table.

Example Get routing instructions and take appropriate action.

Step	Command	Attributes
1	SEND DATA	LINK#> 1 SUBTYPE getrouting VAR LIST A-E ON ERROR, EXECUTE CCT #999
2	RECEIVE DATA	LINK#> 1 ON NAK, EXECUTE CCT # 123 ON ERROR, EXECUTE CCT #999
3	IF	VALUE OF [A] EQ CSR11 EXECUTE CCT #10
4	IF	VALUE OF [A] EQ 1200 EXECUTE CCT #11
5	IF	VALUE OF [A] EQ 999 EXECUTE CCT #12
6	CALL CONTROL TABLE	NUMBER:999

gv*xxxxxxxxxx* Get One VDU Value

The value of data element *xxxxxxxxxx* is retrieved from the EDU Server.

Return Values data_a contains the value returned.

Resp contains "N" for failure and "A" for success.

Example Retrieve the value for "pubcode".

Step	Command	Attributes
1	SEND DATA	LINK#> 1 SUBTYPE gvpubcodeVAR LIST A-E ON ERROR, EXECUTE CCT #999
2	RECEIVE DATA	LINK#> 1 ON NAK, EXECUTE CCT # 123 ON ERROR, EXECUTE CCT #999

newtrans New Transaction

When a NEW TRANSACTION command is executed in a CCT, the call's call reference ID is changed. The newtrans function must then be used to change the call reference ID of the call to match the new ID.

Return Values Resp contains "N" for failure or "A" for success

Example

Step	Command	Attributes
1	NEW TRANSACTION	
2	SEND DATA	LINK#> 1 SUBTYPE newtrans ON ERROR, EXECUTE CCT #999

Note: If you execute a NEW TRANSACTION command in a CCT, you must follow it with a Send Data with a newtrans subtype.

reason*xxx* Reason Code

Stores a data element in the EDU with the name "reason" and a value equal to the contents of data_a.

Return Values This subtype does not return any information.

Example

Step	Command	Attributes
1	SEND DATA	LINK#> 1 SUBTYPE reasoncode011 ON ERROR, EXECUTE CCT #999

startcall Indicate a New Call

This function informs the Legacy Aspect TS that a new call has started, and create an EDU for the call if an EDU does not yet exist. The Legacy Aspect TS sends an event to the Workflow server, passing data from the variables, if any.

data_a contains the call's ANI, while data_b contains the call's DNIS.

Return Values If the Workflow server is programmed to attempt a database hit, the Workflow server returns data values, which are passed to the Aspect CallCenter.

Resp contains "N" for failure and "A" for success.

Errors If an error occurs, an "N" is returned in the Resp field, otherwise an "A" is returned. Errors can occur in a number of ways; for example:

- If the ANI/DNIS database did not have a record for the ANI.
- If the ANI/DNIS database had more than one account number for the given ANI.

Example Send ANI and DNIS information to Avaya IC at the start of a call. Branch to CCT # 123 if the ANI does not represent a unique record.

Step	Command	Attributes
1	MOVE	CONTENTS OF VARIABLE [$] TO VARIABLE [A]
2	MOVE	CONTENTS OF VARIABLE [#] TO VARIABLE [B]
1	SEND DATA	LINK#> 1 SUBTYPE startcall VAR LIST A-E ON ERROR, EXECUTE CCT #999
2	RECEIVE DATA	LINK#> 1 ON NAK, EXECUTE CCT # 123 ON ERROR, EXECUTE CCT #999

sv*xxxxxxxxxx* Set One VDU Value

This function stores a data element in the EDU with a name of *xxxxxxxx* and a value equal to the contents of data_a.

If *xxxxxxxxxx* is "language", the Workflow server will assume that data_a contains a number and attempt to translate it to a language name before placing it in the EDU. The translation is as follows:

data_a	Value placed in EDU
1	English
2	French
3	FrenchCanadian
4	Spanish

If data_a contains a number outside this range or any other text string, it is stored verbatim as a data element in the EDU.

Return Values Resp contains "N" for failure or "A" for success.

Example Store the value for "pubcode".

Step	Command	Attributes
1	SEND DATA	LINK#> 1 SUBTYPE svpubcodeVAR LIST A-E ON ERROR, EXECUTE CCT #999
2	RECEIVE DATA	LINK#> 1 ON NAK, EXECUTE CCT # 123 ON ERROR, EXECUTE CCT #999

INDEX

A

agents
  configuring 35
  monitoring 35
alarm lists and descriptions 67
All 39
aniquery ANI Query 82
annxxx Announcement Played 83
Application Records 30
Aspect CallCenter
  interaction with Avaya Computer Telephony 25
  system requirements 15

C

Call Control Tables
  described 27
  examples 77–79
call flows
  examples 12, 69–75
channel ranges, defined 36
channels, defined 36
Class of Service Records 29
configuration
  agent configuration parameters 35
  TCP/IP 36
containers
  agent 20–21
  call 17–20

D

Data Interlink Records, setting up 23

E

educational services 9
edxxxxxxxxxx Event With Data 84
event descriptions 62–66
evxxxxxxxxxx Event Without Data 84

G

getrouting Get Call Routing Directions 85
gvxxxxxxxxxx Get One eDU Value 86

I

Interface Definition Language (IDL) 40

L

log files 31

M

methods
  background information ??–39
  descriptions of generic TS methods 42–60
  list of generic TS methods 41

N

newtrans New Transaction 86

R

reasonxxx Reason Code 86

S

Send Data command
  described 30
  subtypes 81–88
startcall Indicate a New Call 87
svxxxxxxxxxx Set One eDU Value 88

T

telephony server, defined 11
The 33, 61
Trunk Group Records 26
Trunk Records 26
TS.AnswerVDU 42
TS.Assign 43
TS.AuxWork 62
TS.Busy 44, 62
TS.BusyTerminate 44
TS.ConferenceCancelVDU 45
TS.ConferenceCompleteVDU 45
TS.ConferenceInitVDU 46
TS.Connect 62
TS.Deassign 48
TS.Disconnect 63

TS.Drop 63
TS.GetPhoneInfo 48
TS.HangupVDU 49
TS.Hold 63
TS.HoldReconnect 64
TS.HoldReconnectVDU 50
TS.HoldVDU 50
TS.IncomingCall 64
TS.Login 51, 65
TS.Logout 52, 65
TS.MakeCallTS.MakeCallSetVDU TS.MakeCallVDU 53
TS.Ready 54, 65
TS.ReceiveData 55
TS.Reserved 65
TS.Ring 65
TS.SendData 66
TS.ServerFailed 66
TS.SwapHeld 55
TS.TransferCancelVDU 56
TS.TransferCompleteVDU 57
TS.TransferInitVDU 57
TS.TransferVDU 59
TS.WrapUp 60, 66

U

User Records 28

W

WANs, using the TS in a WAN environment 36
Workflow server
  interaction of TS with 36